



UNIFACS

UNIVERSIDADE SALVADOR

LAUREATE INTERNATIONAL UNIVERSITIES®

MESTRADO EM SISTEMAS E COMPUTAÇÃO

ROBSON GONZAGA SILVA

**PROPOSTA E IMPLEMENTAÇÃO DE MECANISMOS DE SEGURANÇA
APLICADOS AO CONTEXTO DO CAARF-SDN**

Salvador
2019

ROBSON GONZAGA SILVA

**PROPOSTA E IMPLEMENTAÇÃO DE MECANISMOS DE SEGURANÇA
APLICADOS AO CONTEXTO DO CAARF-SDN**

Dissertação apresentada ao Programa de Pós-Graduação em Sistemas e Computação da Universidade Salvador (UNIFACS), *Laureate International Universities*, como requisito parcial para obtenção do título de Mestre.

Orientador: Prof. Dr. Paulo Nazareno Maia Sampaio.

Salvador
2019

Ficha Catalográfica elaborada pelo Sistema de Bibliotecas da UNIFACS Universidade
Salvador, Laureate Internacional Universities.

Silva, Robson Gonzaga.

Proposta e implementação de mecanismos de segurança aplicados ao contexto do CAARF-SDN./ Robson Gonzaga Silva .- Salvador, 2019.

136 f.: il.

Dissertação apresentada ao Programa de Pós-Graduação em Sistemas e Computação da Universidade Salvador (UNIFACS), Laureate International Universities, como requisito parcial para obtenção do título de Mestre.

Orientador: Prof. Dr. Paulo Nazareno Maia Sampaio.

1. Redes de computadores. 2. Segurança da Informação. 3. Segurança em Redes.
I. Sampaio, Paulo Nazareno Maia, orient. II. Título.

CDD: 004.6



UNIFACS
UNIVERSIDADE FEDERAL DO CEARÁ

ATA DE AVALIAÇÃO DE DEFESA DE DISSERTAÇÃO DE MESTRADO

MESTRANDO (A): ROBSON GONZAGA SILVA

TÍTULO: "MODELAGEM E IMPLEMENTAÇÃO DE MECANISMOS DE SEGURANÇA APLICADOS AO CONTEXTO DO CAARE-SDN".

ORIENTADOR (A): PROF. DR. PAULO NAZARENO MAIA SAMPAIO

COMPONENTES DA BANCA:


1. PROF. DR. PAULO NAZARENO MAIA SAMPAIO (ORIENTADOR) - UNIFACS
2. PROF. DR. JOBERTO SÉRGIO BARBOSA MARTINS - UNIFACS
3. PROF. DR. EDUARDO MIGUEL DIAS MARQUES - UMA

COMENTÁRIOS FINAIS:

Dados os trabalhos apresentados, escrito e oral, sua qualidade e relevância, o júri decidiu, por unanimidade, aprovar o candidato, condicionando às correções recomendadas no texto escrito, serem realizadas em um prazo de até 60 dias a partir do dia da defesa.

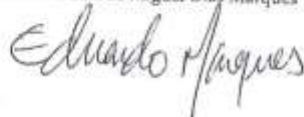
PARECER FINAL: Aprovado

Salvador, 29 de outubro de 2019


Prof. Dr. Paulo Nazareno Maia Sampaio


Prof. Dr. Joberto Sérgio Barbosa Martins

Prof. Dr. Eduardo Miguel Dias Marques



Dedico este trabalho aos meus pais: Nabor e Gleide, que com muita dificuldade ultrapassaram todos limites para me criar e apoiar em minha formação e que me possibilitaram alcançar mais este degrau na minha vida profissional e acadêmica. Dedico também a minha esposa Jessica Thais por toda a paciência e incentivo em todos os momentos dessa caminhada.

Robson Gonzaga Silva

AGRADECIMENTOS

A Deus pela vida e todas as oportunidades.

A todos os docentes do Programa de Pós-Graduação em Sistemas e Computação da UNIFACS que sempre se empenharam ao máximo para proporcionar um excelente ambiente de ensino e a melhor experiência de aprendizagem.

Ao meu professor orientador, Prof. Dr. Paulo Sampaio, pelo profissionalismo e apoio em todas as etapas deste curso, e pela orientação com a qual me conduziu desde o início na elaboração e publicação de artigos em eventos acadêmicos, até a elaboração desta dissertação de mestrado. Também por se mostrar mais que um professor, por se mostrar um grande amigo.

À secretária do Programa de Pós-graduação em Sistemas e Computação da UNIFACS, Sra. Josiane Melo, que cumpriu com todas as responsabilidades proporcionando suporte de forma profissional e ética.

À CAPES por viabilizar minha caminhada nesse curso de mestrado, através da bolsa de estudos e da ajuda de custos que disponibilizou, tornando possível a minha realização acadêmica e profissional.

Aos demais funcionários da Universidade Salvador (UNIFACS), por sempre proporcionarem uma boa recepção, um bom atendimento e por manterem um ambiente agradável para todos os discentes e docentes da instituição.

À minha tia Pretinha e minhas primas, por me apoiarem e proporcionarem hospitalidade em Salvador em todos os momentos que precisei, fornecendo recursos e informações que me ajudaram em toda minha caminhada de estudos em Salvador.

Finalmente, agradeço a minha esposa Jessica Thais pela compreensão, paciência e dedicação, por muitas vezes abrir mão de seus próprios sonhos pessoais para me apoiar nessa caminhada.

RESUMO

A aplicação de redes experimentais através da combinação de diferentes paradigmas, como por exemplo as Redes Sensíveis ao Contexto e Redes Definidas por *Software* (*Software Defined Networking* – SDN), tem se mostrado uma abordagem promissora para o ramo da pesquisa e inovação em redes de computadores devido à combinação de seus principais benefícios como escalabilidade, dinamismo, flexibilidade, fácil gerenciamento, controle de funções, programação e adaptação da rede para atender as mudanças de contexto e as preferências dos usuários. Entretanto, apesar das diversas vantagens advindas com esses novos paradigmas de redes, o estudo da literatura revela alguns desafios de segurança existentes, uma vez que além de introduzir novas vulnerabilidades ao contexto das redes computacionais, tais paradigmas acabam potencializando vulnerabilidades existentes, de uma forma muito mais crítica, aumentando os riscos de exploração das vulnerabilidades por usuários mal-intencionados e criminosos virtuais. Este trabalho tem como objetivo a análise e catalogação de vulnerabilidades e ameaças existentes no contexto híbrido do CAARF-SDN, com o objetivo medir os riscos relacionados aos aspectos de Segurança da Informação (Confidencialidade, Integridade, Disponibilidade, Não Repúdio e Autenticidade). Em um segundo momento, são propostas soluções através do planejamento, implementação e validação de mecanismos de segurança para tratar vulnerabilidades e mitigar ameaças identificadas no contexto do arcabouço CAARF-SDN, aumentando a segurança da rede, assim como de seus recursos e serviços.

Palavras-chave: Redes Definidas por Software – SDN. *Open Network Operating System* – ONOS. Segurança da Informação. Segurança em Redes. DDoS. CAARF-SDN.

ABSTRACT

The application of experimental networks through the combination of different paradigms, such as Context Sensitive Networks and Software-Defined Networks (SDNs), has proved to be a promising approach to the field of research and innovation in computers networks due to the combination of their main benefits such as scalability, dynamism, flexibility, easy management, role control, programming and network adaptation to meet user context changes and preferences. However, despite the various advantages arising from these new paradigms of networks, the literature study reveals some existing security challenges, since in addition to introducing new vulnerabilities to the context of computational networks, that paradigms end up potentiating existing vulnerabilities, in a much more critical way, increasing the risks of exploiting vulnerabilities by malicious users and cybercriminals. This work aims to analyze and catalog vulnerability and threats that exist in the context of the CAARF-SDN hybrid, with the objective of measuring the risks related to the aspects of Information Security (Confidentiality, Integrity, Availability, Non-Repudiation and Authenticity). In a second moment, solutions are proposed through the planning, implementation and validation of security mechanisms to address vulnerabilities and mitigate threats identified in the context of the CAARF-SDN framework, increasing network security, as well as its resources and services.

Keywords: Software Defined Networks – SDN. Open Network Operating System – ONOS. Information Security. Network Security. DDoS. CAARF-SDN.

LISTA DE FIGURAS

Figura 1 - Categorias das informações de contexto.....	28
Figura 2 - Arquitetura de rede convencional X SDN	31
Figura 3 - Arquitetura SDN em camadas	31
Figura 4 – Interação entre o Controlador SDN e o comutador <i>OpenFlow</i>	33
Figura 5 – Implementação Out-of-Band e In-Band.....	34
Figura 6 – Modelo de Contexto do CAARF.....	47
Figura 7 – Controlador de Contexto Projeto CAARF	48
Figura 8 – Arquitetura conceitual do CAARF-SDN	50
Figura 9 – Modelo conceitual do Módulo de Coleta do CAARF-SDN	51
Figura 10 – Modelo conceitual do Módulo de Otimização do CAARF-SDN	53
Figura 11 – Modelo conceitual do Módulo de Atuação do CAARF-SDN.....	54
Figura 12 – Arquitetura Conceitual do Módulo de Integração.....	55
Figura 13 – Fluxograma da Abordagem Operacional do CAARF-SDN.....	57
Figura 14 – Ataque Man In The Middle na rede CAARF-SDN.....	63
Figura 15 – Ataque DDoS na rede CAARF-SDN	65
Figura 16 – Ataque malwares na rede CAARF-SDN.....	67
Figura 17 – Ataque de Falsificação de Switches OpenFlow na rede CAARF-SDN.....	69
Figura 18 – Topologia base dos cenários propostos.....	72
Figura 19 – Ataque SYN Flood no Controlador SDN.....	80
Figura 20 – Modelo conceitual do ataque SYN Flood na rede CAARF-SDN.....	80
Figura 21 – Ataque de ICMP Flood no Controlador SDN	81
Figura 22 – Modelo conceitual do ataque Smurf na rede CAARF-SDN	81
Figura 23 – Captura do Ataque SYN Flood no Wireshark.....	82
Figura 24 – Captura do tráfego na interface de rede do controlador SDN antes do ataque de SYN Flood.....	83
Figura 25 – Captura do tráfego na interface de rede do controlador SDN no momento do ataque de SYN Flood.....	83
Figura 26 – Monitoramento de recursos do Controlador ONOS antes do ataque de SYN Flood	84
Figura 27 – Monitoramento de recursos do Controlador ONOS durante o ataque de SYN Flood	84
Figura 28 – Falha de gerenciamento dos fluxos pelo aplicativo FWD do ONOS.....	85

Figura 29 – Falha do Controlador ONOS na identificação da topologia da rede.....	85
Figura 30 – Falha total na GUI Web do controlador ONOS	86
Figura 31 – Arquitetura conceitual do cenário de tolerância a falhas	87
Figura 32 – Topologia conceitual de implementação de IDS para tráfego espelhado	88
Figura 33 – Topologia conceitual de implementação de IDS inline no ambiente de rede do arcabouço CAARF-SDN	89
Figura 34 – Configuração das regras de alertas maliciosos do Suricata.....	90
Figura 35 – Script do Suricata para configuração e identificação das redes	91
Figura 36 – Script de configuração da localização das regras do Suricata.....	91
Figura 37 – Script de configuração da ferramenta Guardian.....	92
Figura 38 – Ataque SYN Flood direcionado ao Controlador ONOS	93
Figura 39 – Ataque SYN Flood direcionado ao hospedeiro de referência do arcabouço CAARF-SDN	93
Figura 40 – Alertas dos ataques SYN Flood identificados pelo Suricata IDS	94
Figura 41 - Perda de pacotes TCP após a contenção do ataque.....	95
Figura 42 – Análise do ataque SYN Flood no Controlador SDN através da ferramenta Wirehsark.....	95
Figura 43 – Análise da contenção do ataque SYN Flood no Controlador ONOS através da ferramenta Wireshark	96
Figura 44 – Ataque ICMP Flood Smurf direcionado para o hospedeiro de referência do arcabouço CAARF-SDN	97
Figura 45 – Ataque ICMP Flood Smurf direcionado para Controlador ONOS	98
Figura 46 – Análise Wireshark do ataque ICMP Flood Smurf direcionado a rede do CAARF-SDN	98
Figura 47 – Alertas de ataque de Smurf gerados pelo Suricata IDS	99
Figura 48 – Topologia Conceitual do cenário de tolerância a falhas com o Suricata	101
Figura 49 – Interface do Kibana com o gráfico de monitoramento da rede em tempo real baseado nos logs do Suricata	102
Figura 50 – Logs de alertas detalhados gerados pelo Kibana baseados na leitura de arquivos json do Suricata IDS	103
Figura 51 – Modelo Conceitual da arquitetura de cluster para a rede CAARF-SDN	105
Figura 52 – Topologia conceitual do cenário de alta disponibilidade com a visão compartilhada dos Controladores SDN sobre a topologia da rede.....	106

Figura 53 – Topologia conceitual do cenário de alta disponibilidade com a visão compartilhada dos arcabouços Atomix/CAARF-SDN sobre a topologia da rede	107
Figura 54 - Interface do Controlador ONOS com a visão compartilhada da rede e das instâncias do cluster de Controle SDN.....	108
Figura 55 – Detalhes do cluster de controladores na interface do ONOS	108
Figura 56 – Estado do cluster de controladores após falha em uma das instâncias.....	109

LISTA DE GRÁFICOS

Gráfico 1– Análise de gráfico Wireshark da execução do ataque SYN Flood no controlador ONOS no momento do início da contenção do ataque.....	96
Gráfico 2 – Análise de gráfico Wirwshark da execução do ataque SYN Flood no controlador ONOS e do momento exato do bloqueio do ataque	97
Gráfico 3 – Análise da execução do ataque ICMP Flood <i>Smurf</i> no controlador ONOS no momento do início da contenção do ataque.....	99
Gráfico 4 – Análise da execução do ataque ICMP Flood <i>Smurf</i> no controlador ONOS no momento exato do bloqueio do ataque	100
Gráfico 5 – Análise da execução do ataque ICMP Flood <i>Smurf</i> no hospedeiro de referência do arcabouço CAARF-SDN no momento do início da contenção do ataque	100
Gráfico 6 – Análise do ataque ICMP Flood <i>Smurf</i> no hospedeiro de referência do arcabouço CAARF-SDN no momento exato do bloqueio da contenção do ataque	101
Gráfico 7 – Resultado da análise de riscos CVSS no cenário de rede do CAARF-SDN antes da implementação dos mecanismos de segurança.....	111
Gráfico 8 – Resultado da análise de riscos CVSS no cenário de rede do CAARF-SDN após a implementação dos mecanismos de segurança.....	111

LISTA DE TABELAS

Tabela 1 – Vulnerabilidade e ameaças em SDN	61
Tabela 2 - Descrição da topologia base dos experimentos de rede	73
Tabela 3 - Configuração de <i>hardware</i> dos dispositivos da topologia	74
Tabela 4 – Aplicações utilizadas	78

LISTA DE ACRÔNIMOS

ABNT	Associação Brasileira de Normas Técnicas
API	<i>Application Programming Interface</i>
ARP	<i>Address Resolution Protocol</i>
AS	<i>Autonomous System</i>
BGP	<i>Border Gateway Protocol</i>
CAARF-SDN	<i>Context-Aware Adaptive Routing Framework – Software Defined Network</i>
CE	<i>Contextual Elements</i>
CERT.BR	Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil
CLI	<i>Comand-Line Interface</i>
CPU	<i>Central Processing Unit</i>
CVSS	<i>Common Vulnerability Scoring System</i>
DDoS	<i>Distributed Denial of Service</i>
DoS	<i>Denial of Service</i>
FWD	<i>Forward</i>
GNS3	<i>Graphical Network Simulator 3</i>
GUI	<i>Graphical User Interface</i>
HD	<i>Hard Disk</i>
HID	<i>Hybrid Intrusion Detection</i>
HIDS	<i>Host-based Intrusion Detection System</i>
HIPS	<i>Host-based Intrusion Prevention System</i>
HTTP	<i>Hypertext Transfer Protocol</i>
ICMP	<i>Internet Control Message Protocol</i>
IDS	<i>Intrusion Detection Systems</i>
IoT	<i>Internet of Things</i>

IP	<i>Internet Protocol</i>
IPS	<i>Intrusion Prevention Systems</i>
JSON	<i>JavaScript Object Notation</i>
LAN	<i>Local Area Network</i>
MITM	<i>Man in the Middle</i>
NIDS	<i>Network Intrusion Detection System</i>
NIPS	<i>Network Intrusion Prevention System</i>
NOS	<i>Network Operating System</i>
ONF	<i>Open Networking Foundation</i>
ONOS	<i>Open Network Operating System</i>
OSPF	<i>Open Shortest Path First</i>
POO	<i>Programação Orientada a Objetos</i>
PSI	<i>Política de Segurança da Informação</i>
QoC	<i>Quality of Context</i>
QoD	<i>Quality of Device</i>
QoE	<i>Quality of Experience</i>
QoS	<i>Quality of Service</i>
RAM	<i>Random Access Memory</i>
RIP	<i>Routing Information Protocol</i>
SDN	<i>Software Defined Network</i>
SHADE	<i>Selective High Rate DDoS Defense</i>
SLA	<i>Service Level Agreement</i>
SNMP	<i>Simple Network Management Protocol</i>
SSL	<i>Secure Sockets Layer</i>
TCP	<i>Transmission Control Protocol</i>
TLS	<i>Transport Layer Security</i>
UDP	<i>User Datagram Protocol</i>

VoIP

Voice Over Internet Protocol

WAN

Wide Area Network

SUMÁRIO

1	INTRODUÇÃO.....	20
1.1	PROBLEMÁTICA	21
1.2	MOTIVAÇÕES	22
1.3	OBJETIVOS	22
1.4	METODOLOGIA	23
1.5	PRINCIPAIS CONTRIBUIÇÕES.....	24
1.6	ORGANIZAÇÃO DA DISSERTAÇÃO.....	24
2	REVISÃO CONCEITUAL DE LITERATURA	26
2.1	INTRODUÇÃO	26
2.2	CONCEITOS E DEFINIÇÕES	26
2.2.1	Qualidade de Serviço	26
2.2.2	Qualidade de Experiência	27
2.2.3	Qualidade de Dispositivos	27
2.2.4	Contexto	28
2.3	REDES DEFINIDAS POR SOFTWARE	30
2.3.1	Protocolo <i>OpenFlow</i>	32
2.3.2	Controlador	33
2.4	SEGURANÇA EM REDES DE COMPUTADORES	34
2.4.1	Principais Ameaças em Redes de Computadores	35
2.4.2	Aspectos de Segurança em Redes – Principais Pilares	37
2.4.3	Mecanismos de Segurança	38
2.5	TRABALHOS RELACIONADOS – SEGURANÇA EM REDES SDN ...	41
2.6	CONCLUSÃO	45
3	CAARF-SDN: ARCABOUÇO BASEADO EM CONTEXTO.....	47
3.1	INTRODUÇÃO	47

3.2	MODELO DE CONTEXTO	47
3.3	ARQUITETURA DO CAARF-SDN.....	48
3.3.1	Módulo de Coleta	51
3.3.2	Módulo de Otimização	52
3.3.3	Módulo de Atuação	54
3.3.4	Módulo de Integração	55
3.4	GESTÃO DE ENCAMINHAMENTO DO CAARF-SDN	57
3.5	CONCLUSÃO	58
4	ASPECTOS DE SEGURANÇA DA INFORMAÇÃO APLICADOS AO ARCABOUÇO CAARF-SDN	60
4.1	INTRODUÇÃO	60
4.2	VULNERABILIDADES E AMEAÇAS NA ARQUITETURA CAARF-SDN	60
4.2.1	Ataques <i>Man In The Middle</i> no CAARF-SDN	62
4.2.2	Negação de Serviço (<i>Denial of Service - DoS</i>) no CAARF-SDN	64
4.2.3	<i>Softwares Maliciosos (Malwares)</i> no CAARF-SDN	66
4.2.4	Ataques de Falsificação de <i>Switch OpenFlow</i> no CAARF-SDN	68
4.3	CONCLUSÃO	69
5	SOLUÇÕES PROPOSTAS	71
5.1	INTRODUÇÃO	71
5.2	CARACTERIZAÇÃO DO EXPERIMENTO	71
5.2.1	Topologia Utilizada nos Cenários	72
5.2.2.	Definição das Aplicações Utilizadas	75
5.3	CARACTERIZAÇÃO DOS ATAQUES DoS (<i>DENIAL of SERVICE</i>) REALIZADOS	79
5.4	CENÁRIO 1 - ATAQUES DoS SEM MECANISMOS DE CONTENÇÃO.	79
5.4.1	Ataques de SYN Flood e ICMP Flood na Rede CAARF-SDN	80
5.5	CENÁRIO 2 – TOLERÂNCIA A FALHAS NO CONTEXTO CAARF SDN	86

5.5.1	Solução de Tolerância a Falhas na rede CAARF-SDN	87
5.5.2	Contenção de Ataques SYN Flood na Rede do Arcabouço CAARF-SDN	93
5.5.3	Contenção de Ataques ICMP Flood no Contexto do CAARF-SDN	97
5.6	CENÁRIO 3 - ALTA DISPONIBILIDADE NA REDE CAARF-SDN ...	103
5.7	DISCUSSÃO E LIÇÕES APRENDIDAS	109
5.8	CONCLUSÃO	112
6	CONCLUSÕES E PERSPECTIVAS FUTURAS	113
6.1	CONCLUSÕES	113
6.2	PERSPECTIVAS FUTURAS.....	116
	REFERÊNCIAS	124
	APÊNDICE A - CONFIGURAÇÃO REGRAS DO SURICATA.....	131
	APÊNDICE B - CONFIGURAÇÃO BÁSICA DO SURICATA IDS.....	132
	APÊNDICE C - CONFIGURAÇÃO DO GUARDIAN	134
	APÊNDICE D - CONFIGURAÇÃO DE INTEGRAÇÃO LOGSTACK	135
	APÊNDICE E - CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 1	137
	APÊNDICE F - CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 2	138
	APÊNDICE G - CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 3.....	139
	APÊNDICE H - CONFIGURAÇÃO DO ONOS INSTÂNCIA 1	140
	APÊNDICE I - CONFIGURAÇÃO DO ONOS INSTÂNCIA 2.....	141
	APÊNDICE J - CONFIGURAÇÃO DO ONOS INSTÂNCIA 3	142

1 INTRODUÇÃO

A demanda por uma gestão otimizada dos recursos disponíveis da infraestrutura de rede vem crescendo exponencialmente, a fim de lidar com a coexistência do crescente tráfego heterogêneo na Internet e das diversas tecnologias envolvidas. Portanto, como a Internet não foi projetada para atender a essa demanda, questões como vulnerabilidade, instabilidade, escalabilidade e incompatibilidades também são mais evidentes. O sucesso e crescimento da Internet é inegável, porém algumas de suas limitações estão sendo desveladas, por isso é importante reconsiderar sua arquitetura e principais protocolos com novas redes experimentais.

Os esforços realizados nos últimos anos para propor inovações e alterações na arquitetura original das redes computacionais não promoveram grandes mudanças em sua estrutura básica (GOMES; ISHIMORI; FARIAS, 2013). Muitos problemas surgiram nessa arquitetura legada ao longo do tempo devido a sua grande complexidade e rigidez, dificultando o processo de inovação e pesquisa (MOREIRAS, 2016).

Diversas soluções vêm sendo propostas ao longo dos anos buscando promover maior flexibilidade e proporcionar ambientes de redes mais eficientes (FERREIRA, 2016). Dessa forma, afim de melhorar a experiência de uso e incorporar o conceito de rede centrada no usuário, a noção de contexto pode ser aplicada, onde a consciência de contexto é entendida como um paradigma computacional onipresente e difundido, ao qual o ambiente computacional reage de acordo com as mudanças em seu estado atual (SHAIKH; FIEDLER; COLLANGE, 2010). A implementação de Redes Sensíveis ao Contexto (*Context-Aware Network – CAN*) (WU et al., 2014) pode ser útil para melhorar a satisfação do usuário em relação a sua percepção ao acessar recursos de rede e no aprimoramento do gerenciamento de tráfego, levando em consideração aspectos do usuários, da rede e dos dispositivos finais, oferecendo uma abordagem genérica e de ponta para otimização de tráfego.

Outro paradigma importante no contexto das novas tecnologias de infraestrutura de rede e telecomunicação, são as Redes Definidas por *Software* (*Software Defined Network – SDN*) que fornecem mecanismos para a implementação de controle dinâmico e gerenciamento de recursos de rede para entrega de tráfego heterogêneo, através da separação do plano de controle e plano de dados e a capacidade de programação do comportamento da rede (KIM; FEAMSTER, 2013). No entanto, por padrão (*default*) a configuração da tabela de fluxo (*flowtable*) dentro dos controladores SDN é realizado estaticamente. Dessa forma, visando uma alternativa para otimização de tráfego de rede e configuração dinâmica das tabelas de fluxo em SDN, foi proposta na literatura uma solução de otimização centrada no usuário (baseada em

contexto) chamada *Context-Aware Adaptive Routing Framework* (SPINOLA, 2015) aplicada a redes SDN (também chamada de CAARF-SDN) (BADARÓ NETO et al., 2018). A proposta híbrida do CAARF-SDN, é formada pela integração dos conceitos de Qualidade de Serviço (*Quality of Service - QoS*), Qualidade da Experiência (*Quality of Experience - QoE*) e Qualidade de Dispositivo (*Quality of Device - QoD*) para oferecer uma abordagem mais proativa e dinâmica ao tráfego sensível ao tempo (como VoIP e vídeo), visando a melhoria da percepção do usuário sobre redes IP convencionais ou experimentais (OLIVEIRA et al., 2015).

Através da combinação de aspectos das Redes Sensíveis ao Contexto e SDN em um modelo híbrido, é possível somar os benefícios de ambos os paradigmas, como escalabilidade, dinamismo, flexibilidade, fácil gerenciamento e controle de funções, programação e adaptação de parâmetros e comportamento da rede de acordo com as mudanças do contexto (BADARÓ NETO et al., 2018).

Apesar das vantagens e possibilidades encontradas na abordagem híbrida do CAARF-SDN, a junção desses paradigmas acabam introduzindo vulnerabilidades e ameaças inéditas ao contexto das redes de computadores, além de potencializar ameaças e vulnerabilidades já existentes em abordagens tradicionais ou legadas (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016).

Dessa maneira, a proposta da presente dissertação é promover a implementação de mecanismos de segurança de rede aplicados ao arcabouço CAARF-SDN, visando garantir aspectos fundamentais de segurança, aumentando a disponibilidade dos recursos da rede e diminuindo as possibilidades de ocorrências de falhas relacionadas a tentativas de intrusão. Para isso, buscou-se identificar, analisar e catalogar vulnerabilidades e ameaças existentes no contexto de rede do CAARF-SDN, afim de orientar a implementação de forma alinhada com as principais necessidades de segurança do arcabouço.

1.1 PROBLEMÁTICA

Considerando a contextualização apresentada, foram identificados os principais problemas que direcionam a proposta deste trabalho de mestrado:

- a) Existência de vulnerabilidades críticas e não tratadas em ambientes de redes baseados em novos paradigmas, tais como o CAARF-SDN;

- b) Ausência de mecanismos para prevenir intrusão e falhas no arcabouço CAARF-SDN, e;
- c) Ausência de planejamento focado na implementação de alta disponibilidade para o arcabouço CAARF-SDN.

1.2 MOTIVAÇÕES

Tendo em vista a relevância dos novos paradigmas de redes de computadores para a comunidade científica, tais como SDN e as Redes Sensíveis ao Contexto, assim como a importância das propostas híbridas como alternativas para suprir as novas demandas de serviços de rede, mostra-se cada vez mais necessário propor soluções em resposta aos problemas de segurança existentes nesse contexto convergente de novas tecnologias.

Portanto, faz-se necessário o desenvolvimento de estudos orientados aos aspectos de segurança no contexto desses novos paradigmas de redes de computadores, a fim de estabelecer princípios como Confidencialidade, Integridade e Disponibilidade aos ativos e recursos da rede.

Dessa forma, a presente dissertação tem como motivação a busca por soluções de segurança, tendo como foco mitigar as principais vulnerabilidades e ameaças identificadas no contexto dos novos paradigmas de redes, mais especificamente na proposta híbrida entre Redes Definidas por *Software* e Redes Sensíveis ao Contexto.

1.3 OBJETIVOS

O trabalho realizado na presente dissertação de mestrado tem por objetivo geral apresentar a aplicação de estratégias e implementação de mecanismos de segurança no contexto de rede do arcabouço CAARF-SDN, levando em consideração as boas práticas para a mitigação de ameaças, buscando garantir maior tolerância a falhas contra ataques relacionados a vulnerabilidades de intrusão de rede e ainda a implementação de recursos para provisionar maior disponibilidade dos recursos e ativos da rede e do arcabouço CAARF-SDN.

Desta forma, o objetivo geral deste estudo foram divididos nos seguintes objetivos específicos:

- a) Identificar e catalogar as principais vulnerabilidades e ameaças existentes no contexto de rede do arcabouço CAARF-SDN;

- b) Propor soluções de segurança baseados nas necessidades de segurança do arcabouço CAARF-SDN;
- c) Propor estudos de casos para viabilizar a implementação e validação das soluções propostas;
- d) Implementar as soluções de segurança propostas visando garantir maior tolerância a falhas e garantir alta disponibilidade dos recursos e ativos da rede e do arcabouço CAARF-SDN, e;
- e) Apresentar os resultados de avaliação e validação dos mecanismos de segurança implementados nos experimentos.

1.4 METODOLOGIA

A presente pesquisa se desenvolveu em primeiro momento por meio da busca e análise de literatura pertinente a temática abordada, onde foram consultadas diversas fontes como sites, revistas científicas, artigos publicados e livros digitais, buscando ampliar a capacidade crítica sobre o assunto. Em segundo momento, foram realizadas análises na documentação do CAARF-SDN, buscando identificar e catalogar as principais vulnerabilidades e ameaças relacionadas ao arcabouço. Por fim, foram implementados alguns cenários baseados em virtualização para realização dos experimentos e validação das soluções propostas.

Quanto a sua natureza, a presente dissertação caracteriza-se como pesquisa do tipo aplicada, uma vez que busca gerar conhecimento com o intuito de propor soluções práticas para resolver problemas específicos (GIL, 2008).

No que diz respeito a abordagem, o tipo de pesquisa apresenta-se como quantitativa por manter seu foco em analisar os problemas relacionados ao objeto de estudo de forma estatística, a fim de mensurar as possibilidades, padrões e resultados dentro dos experimentos realizados (SILVA; MENEZES, 2005).

Acerca dos objetivos, essa pesquisa define-se como exploratória, por proporcionar maior familiaridade com o problema em questão, de forma a torná-lo explícito. Também apresenta características do tipo explicativa, uma vez que tem por objetivo identificar fatores que determinam ou contribuem com a ocorrência do problema (GIL, 2008).

Em relação aos procedimentos técnicos, essa dissertação apresenta características de pesquisa bibliográfica por ser elaborada a partir da consulta de materiais da literatura, como foi apresentado inicialmente. Possui ainda características de pesquisa experimental, uma vez

que tem por objetivo reproduzir os fenômenos estudados em ambientes controlados, afim de evidenciar a relação existente entre os fatos e a teoria (SILVA; MENEZES, 2005).

1.5 PRINCIPAIS CONTRIBUIÇÕES

Como contribuição do trabalho realizado na presente dissertação estão as discussões relacionadas à segurança em redes computacionais no contexto dos novos paradigmas e principalmente no projeto CAARF-SDN, possibilitando a identificação e catalogação de vulnerabilidades e ataques críticos no ambiente de rede do arcabouço.

Outra contribuição do presente trabalho é a análise de riscos desenvolvida tomando como base as principais métricas de segurança em redes, entretanto de forma personalizada de acordo com as necessidades do arcabouço CAARF-SDN, permitindo medir e avaliar o impacto de ataques e a eficiência das soluções propostas no contexto de rede do arcabouço.

Ainda como contribuição direta, foi proposto um ambiente de experimentação totalmente viável para nortear futuras pesquisas relacionadas ao arcabouço CAARF-SDN, tendo em vista que algumas ferramentas utilizadas nos últimos experimentos do arcabouço tornaram-se obsoletas e inviáveis.

Por fim, o presente trabalho contribuiu como extensão das pesquisas do CAARF-SDN, modelando, implementando e validando mecanismos de segurança no contexto de rede do CAARF-SDN, promovendo um ambiente mais confiável, íntegro e tolerante a falhas.

1.6 ORGANIZAÇÃO DA DISSERTAÇÃO

Além deste capítulo de introdução, esta dissertação está organizada em mais cinco capítulos, conforme apresentados a seguir:

- a) Capítulo 2 – Revisão conceitual e de literatura: Nesse capítulo é apresentada uma revisão conceitual e literária relacionada aos principais aspectos envolvidos no estudo desta dissertação, além da identificação e estudo dos principais trabalhos análogos existentes na literatura;
- b) Capítulo 3 – CAARF-SDN – Arcabouço Baseado em Contexto: neste capítulo são apresentados de forma geral, os conceitos e características da nova arquitetura do *Context-aware Adaptive Routing Framework* (CAARF) sob o aspecto do paradigma das Redes Definidas por Software, denominado de CAARF-SDN;

- c) Capítulo 4 – Aspectos de Segurança da Informação Aplicados ao CAARF-SDN: Nesse capítulo é apresentado o estudo relacionado aos principais aspectos de segurança voltados para o ambiente de redes do arcabouço CAARF-SDN, como a identificação e catalogação de vulnerabilidades, levantamento e análise de riscos e principais tipos de ataques;
- d) Capítulo 5 – Soluções Propostas e Experimentos: É apresentado nesse capítulo a estruturação e resultados do experimento realizado para validação das estratégias e mecanismos de segurança apresentados como parte da proposta dessa dissertação, e;
- e) Capítulo 6 – Conclusões e Perspectivas Futuras: Nesse capítulo são apresentadas as principais conclusões relacionadas ao presente trabalho, assim como as perspectivas para trabalhos futuros.

2 REVISÃO CONCEITUAL DE LITERATURA

2.1 INTRODUÇÃO

Neste capítulo são apresentados os principais conceitos relacionados direta ou indiretamente ao tema desta dissertação de mestrado, através de uma revisão conceitual da literatura baseada em aspectos das redes computacionais como qualidade de serviço (*Quality of Service – QoS*), qualidade de experiência (*Quality of Experience - QoE*), qualidade de dispositivo (*Quality of Device – QoD*), engenharia de tráfego, abordagens baseadas em contexto, redes definidas por software (*Software Defined Network – SDN*) e segurança em redes de computadores.

Todos os estudos realizados para o desenvolvimento deste trabalho foram direcionados pelos temas citados, com o objetivo de auxiliar o processo de desenvolvimento e adaptação de soluções, para implementação e validações dos mecanismos propostos nesta dissertação de mestrado.

2.2 CONCEITOS E DEFINIÇÕES

Nessa seção são descritos os principais conceitos e definições utilizados no desenvolvimento desse trabalho.

2.2.1 Qualidade de Serviço

Os constantes avanços das redes de computadores ocasionaram a explosão do processo de produção e consumo de conteúdos multimídia (voz e vídeo) na *Internet*. Esses tipos de conteúdo necessitam de uma abordagem mais criteriosa, envolvendo o tratamento de parâmetros mais sensíveis e específicos da rede, tais como confiabilidade (*reliability*), largura de banda (*bandwidth*), atraso (*delay*), perda (*loss*), variação do atraso (*jitter*), que juntos estabelecem requisitos de definição de Qualidade de Serviço (*Quality of Service – QoS*) (TANENBAUM, 2007; BON, 2008; SHAIKH; FIEDLER; COLLANGE, 2010).

Dessa forma, a qualidade de serviço tem o objetivo de atender às expectativas dos usuários em relação aos serviços, buscando a melhoria na qualidade da comunicação e no tempo de resposta das aplicações da rede (WU et al., 2009)C.

O termo QoS também é utilizado na identificação de aspectos e técnicas que vão além do desempenho da rede, estando relacionado ainda com requisitos de “segurança, privacidade,

contabilidade, política de preços dos serviços, estabelecimento e monitoração de contratos de serviços e grau de disponibilidade da rede.” (MUAKAD, 2015).

2.2.2 Qualidade de Experiência

A qualidade de experiência (*Quality of Experience – QoE*) é considerada um conceito abstrato que utiliza informações do usuário para criar garantias que refletem na qualidade de serviço. Tem por objetivo, mensurar todos os fatores subjetivos e objetivos na percepção da qualidade pelo o usuário em relação a sua interação com sistemas, produtos e serviços, abrangendo fatores cognitivos, comportamentais, psicológicos, contextuais, tecnológicos e comerciais (LAGHARI; CONNELLY, 2012). Esses fatores são apresentados na literatura em três grupos principais: (i) Fatores de influência humana (sexo, humo, expectativas, personalidade, [...]), (ii) fatores de influência do sistema (largura de banda, atraso, resolução de tela, e (iii) fatores de influência do contexto (localização, espaço, tempo, frequência de uso, [...]) (BRUNNSTRÖM et al., 2014).

Apesar do QoE ser percebido de forma subjetiva (LAGHARI et al., 2011), é uma medida importante no processo de auto avaliação das operadoras em relação ao nível de satisfação dos usuários para com os serviços prestados, ajudando a elevar a qualidade destes serviços.

2.2.3 Qualidade de Dispositivos

A qualidade de dispositivo (*Quality of Devices - QoD*) está relacionada com as informações e características técnicas do hardware (memória, processador, bateria, [...]) dos dispositivos de comunicação, sendo de suma importância na avaliação do uso adequado da rede (BADARÓ NETO et al., 2018). Através do QoD é possível mensurar o desempenho do hardware e avaliar se determinado dispositivo atende adequadamente as demandas impostas pelas necessidades dos sistemas ou serviços (OLIVEIRA et al., 2015). Sendo assim, é possível medir métricas relevantes de um dispositivo como tempo de resposta, precisão, temperatura, tolerância a falhas e dentre outros.

Dessa forma é possível avaliar a qualidade de um determinado dispositivo dentro de um ponto de vista global (contexto) do sistema, que resulta no impacto direto ao nível de qualidade de serviço percebida pelo usuário, assim como em sua experiência de uso do sistema (NAZARIO; DANTAS; TODESCO, 2012).

2.2.4 Contexto

Com a evolução da tecnologia móvel, da computação ubíqua e pervasiva, assim como o advento da Internet das Coisas (*Internet of Things* – IoT), ampliou-se o conceito de interação do usuário com os sistemas computacionais. Para Muakad, (2015, p. 26, apud REN; SENG, 2009), “este tipo de computação apresenta forte ligação” com o mundo físico, uma vez que tenta representar as características dos elementos do mundo real. As informações utilizadas para representar essas características são denominadas de contexto (MUAHAD, 2015).

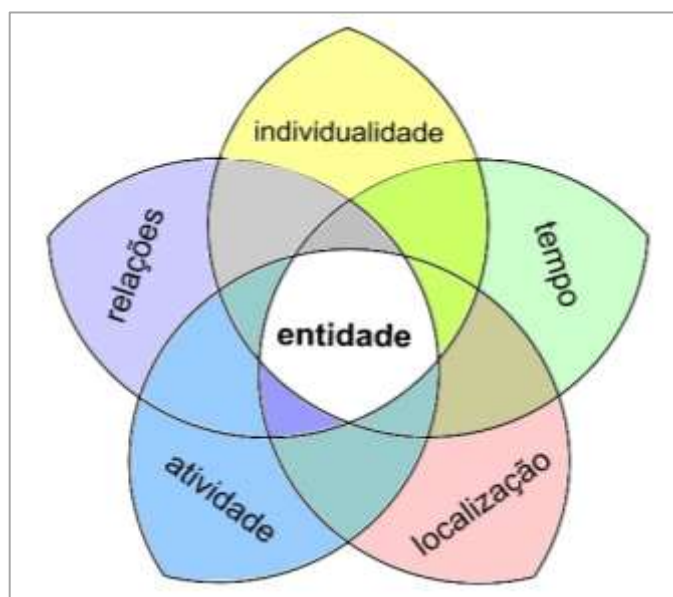
Portanto, contexto pode ser definido como qualquer informação que possa caracterizar uma situação relacionada a uma entidade (pessoas, lugares, objetos e etc.), e que seja relevante para interação entre um usuário e uma aplicação (DEY; ABOWD; SALBER, 2001).

A definição do contexto tem o objetivo de descrever os Elementos Contextuais (*Contextual Elements* – CEs), caracterizando o domínio no qual uma entidade está inserida (VIEIRA; TEDESCO; SALGADO, 2009).

O contexto de uma entidade é caracterizado através do relacionamento entre suas respectivas informações contextuais, que são classificadas em cinco categorias bem definidas. Essas categorias são utilizadas para descrever e detalhar as informações que definem um contexto, a saber: individualidade, tempo, atividade, localização e relações (ZIMMERMANN; LORENZ; OPPERMANN, 2007).

A Figura 1 ilustra a interação entre as categorias de informações que definem o contexto de uma entidade.

Figura 1 - Categorias das informações de contexto



Fonte: Zimmermann, Lorenz e Oppermann (2007).

Tais categorias estabelecem a consciência do contexto, que pode ser definida como a capacidade de adaptação de um sistema em relação as mudanças de contexto onde está inserido:

- a) **Contexto de Individualidade (*Individuality*):** Representa as características particulares de uma entidade, possibilitando a sua identificação de forma única. Descreve informações relacionadas a interações com o ambiente, comportamento humano, interações com objetos e agrupamento de entidades (ZIMMERMANN; LORENZ; OPPERMANN, 2007);
- b) **Contexto de Tempo (*Time*):** Essa categoria identifica as informações relacionadas ao uso do tempo dentro de um contexto, como fuso horário de um determinado cliente ou ainda o tempo virtual em algum sistema por exemplo. O tempo é uma variável fundamental para avaliar mudanças de comportamento e no ambiente ao longo de um período temporal, possibilitando o armazenamento, acesso e análise de históricos de interações e comportamentos humanos ou de objetos dentro de um determinado contexto, servindo ainda para inferir ações e comportamentos futuros, influenciando diretamente no processo de tomada de decisões (ZIMMERMANN; LORENZ; OPPERMANN, 2007);
- c) **Contexto de Localização (*Location*):** As informações apresentadas nesse contexto descrevem modelos de localização que podem representar o espaço físico ou virtual de uma entidade, assim como parâmetros mais específicos como velocidade e orientação (ZIMMERMANN; LORENZ; OPPERMANN, 2007). Percebe-se que o contexto de localização mantém constantes relações com o contexto de tempo, uma vez que parâmetros como velocidade, posicionamento geográfico, medição de ângulos e entre outros, apresenta fortes características temporais (ZIMMERMANN; LORENZ; OPPERMANN, 2007);
- d) **Contexto de Atividade (*Activity*):** As atividades no contexto de uma entidade, descrevem seus objetivos e ações referentes a um grupo de tarefas desempenhadas pela mesma. Essa categoria abrange todas as atividades que uma entidade executa no tempo presente e ainda no futuro, apresentando seus objetivos e metodologias utilizadas para se alcançar tais objetivos, que podem ser

descritas através de tarefas, objetivos e ações explícitas (ZIMMERMANN; LORENZ; OPPERMANN, 2007), e;

- e) **Contexto de Relações:** Essa categoria abrange informações contextuais obtidas através das interações estabelecidas entre as entidades pertencentes a determinado contexto. Dessa forma, as entidades estabelecem uma relação estática ou dinâmica de vizinhança e o resultado dessas interações gera um conjunto de todas as relações da entidade (ZIMMERMANN; LORENZ; OPPERMANN, 2007).

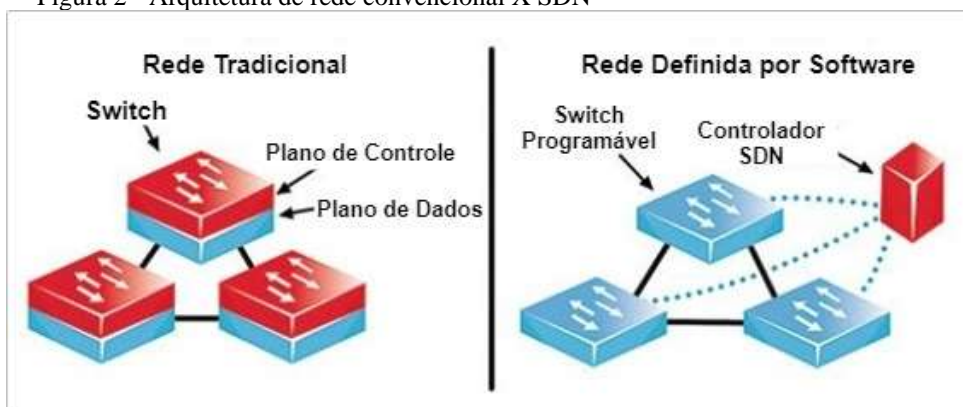
2.3 REDES DEFINIDAS POR SOFTWARE

O termo SDN (*Software-Defined Networking*) foi inicialmente proposto para representar conceitos relacionados ao projeto do protocolo *OpenFlow*, sendo que algumas das primeiras propostas desse paradigma surgiram no ano de 2008 em pesquisas desenvolvidas entre a Universidade de Stanford e a Universidade da Califórnia em Berkeley, com o objetivo de proporcionar uma plataforma aberta para a comunidade de pesquisa relacionada à tecnologia e novas soluções para redes de computadores (MCKEOWN et al., 2008).

A principal característica das redes SDN consiste na separação física entre o plano de dados e o plano de controle, viabilizando a centralização lógica da inteligência da rede. Nessa arquitetura, um novo componente denominado controlador é inserido no plano de controle com o objetivo de definir uma interface de programação, possibilitando a manipulação das ações dos comutadores da rede (switches, roteadores, entre outros) em relação ao encaminhamento de pacotes (KREUTZ et al., 2014). Essa abordagem garante a interoperabilidade dos dispositivos e facilita o gerenciamento da rede, dessa forma, a implementação de novas funcionalidades de rede torna-se independente do hardware do fabricante e aumenta o controle sobre a rede (UNDERDAHL; KINGHORN, 2015). Essas características fazem das redes SDN um novo paradigma de redes de computadores, uma vez que em ambientes legados, os planos de dados e controle coexistem no mesmo dispositivo e são gerenciados através de um sistema operacional embarcado que não possibilita a extensão das funcionalidades definidas pelo fabricante (XIA et al., 2014).

A Figura 2 mostra a comparação entre uma rede convencional e uma rede baseada na arquitetura SDN.

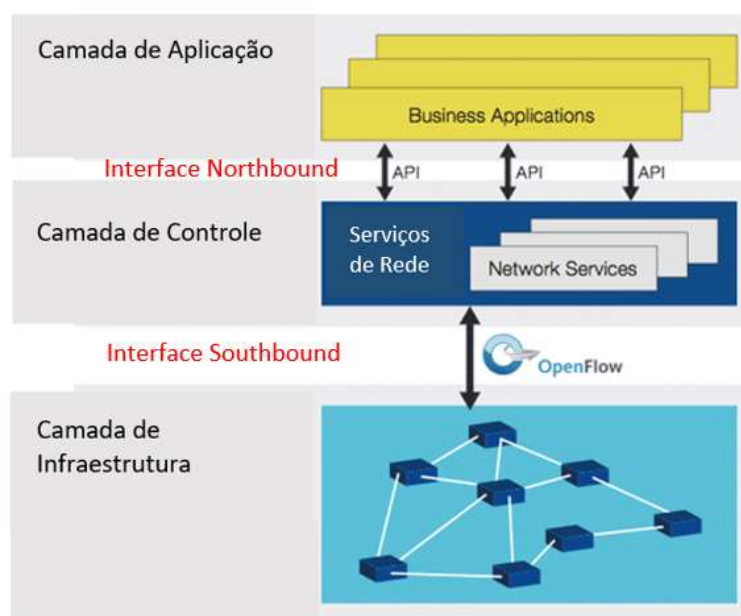
Figura 2 - Arquitetura de rede convencional X SDN



Fonte: Dungay (2016).

Para Kreutz et al. (2014), a separação do plano de controle e do plano de dados permite o roteamento explícito e a inspeção do tráfego de rede, possibilitando o controle dos elementos de forma mais eficiente em comparação com as redes tradicionais. Basicamente, a arquitetura das redes SDN são classificadas em três camadas, encaminhamento de dados, controle e aplicação (YOON et al., 2017; SHU et al., 2016; ONF, 2013). A Figura 3 ilustra a visão em camadas da arquitetura SDN.

Figura 3 - Arquitetura SDN em camadas



Fonte: ONF (2013).

A camada de encaminhamento de dados, ou camada de infraestrutura consiste em vários dispositivos de encaminhamento interligados entre si e com o controlador. Os vários

dispositivos estão fisicamente conectados por mídias com ou sem fio e são simplesmente responsáveis por encaminhar pacotes, baseados em regras armazenados em suas tabelas internas. É importante destacar que a interação entre essa camada e a de controle é realizada através de uma interface denominada de *southbound* por meio do protocolo *OpenFlow* (SHU et al., 2016).

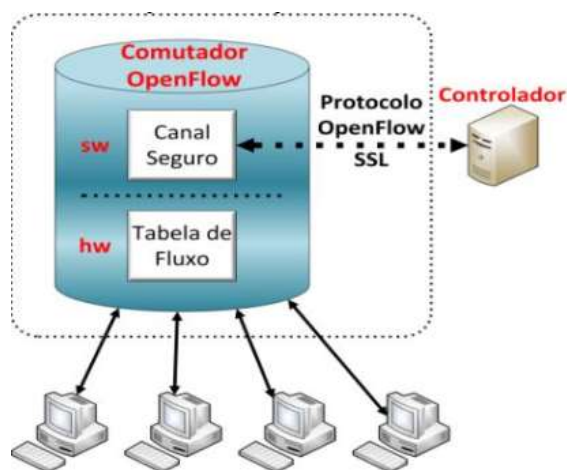
A camada de controle é responsável por aplicar toda a lógica de rede através de um ou mais elementos de controle, gerenciando e controlando o comportamento geral de dispositivos e do tráfego, por meio de políticas bem definidas e regras de fluxo (SHU et al., 2016). Dessa forma, "a função de camada de controle é análoga a de um sistema operacional tradicional executando programas" e gerenciando aplicativos SDN (YOON et al., 2017).

A camada de aplicativo permite a programação de aplicações para controle de rede. Nesta camada a programação pode ser feita "através de *Application Programming Interface* (API), *scripts*, ou qualquer outro ambiente que em termos de *software* permite ao cliente usar rotinas estabelecidas e padrões" para realizar a administração e gestão da rede de forma transparente, abstraindo detalhes técnicos ou de implementação. Nesse aspecto, a interação dos aplicativos desta camada com o controlador SDN é realizada por meio de uma interface denominada *northbound* (MOREIRAS, 2016).

2.3.1 Protocolo *OpenFlow*

O protocolo *OpenFlow* é definido como um padrão aberto padronizado pelo ONF (*Open Network Foundation*), que habilita a interface *southbound* para comunicação entre os comutadores *OpenFlow* (camada de infraestrutura) e o controlador (camada de controle), por meio de um canal seguro baseado em criptografia TLS/SSL (*Transport Layer Secure /Secure Socket Layer*), permitindo a programação das tabelas de fluxo dos comutadores, sendo a tabela de fluxo uma caracterização dos fluxos de rede recebidos por esses comutadores *OpenFlow* (MCKEOWN et al., 2008).

A Figura 4 ilustra a interação entre o controlador SDN e um comutador *OpenFlow* através do canal TLS/SSL utilizando o protocolo *OpenFlow*.

Figura 4 – Interação entre o Controlador SDN e o comutador *OpenFlow*

Fonte: ONF (2013).

É importante destacar que mesmo sendo conceitos relacionados, SDN e *OpenFlow* têm papéis diferentes, sendo que o *OpenFlow* atua apenas na definição de um conjunto de instruções, e através destas, possibilita o controle genérico do encaminhamento de pacotes da rede, por meio da manipulação das ações de comutadores que estejam habilitados para executar o protocolo, enquanto que o paradigma SDN é um conceito mais abrangente em relação a arquitetura das redes através de uma nova perspectiva, que consiste na separação do plano de dados e plano de controle.

2.3.2 Controlador

No contexto das Redes Definidas por *Software*, o controlador é definido como um elemento logicamente centralizado dotado da capacidade de manipular o comportamento da rede como um todo. É "uma entidade de *software* que tem controle exclusivo sobre um conjunto abstrato de recursos de plano de dados." (MING et al., 2015). Dessa forma, é o controlador que fornece a interface de programação que possibilita o desenvolvimento de aplicativos e funções para controle da rede.

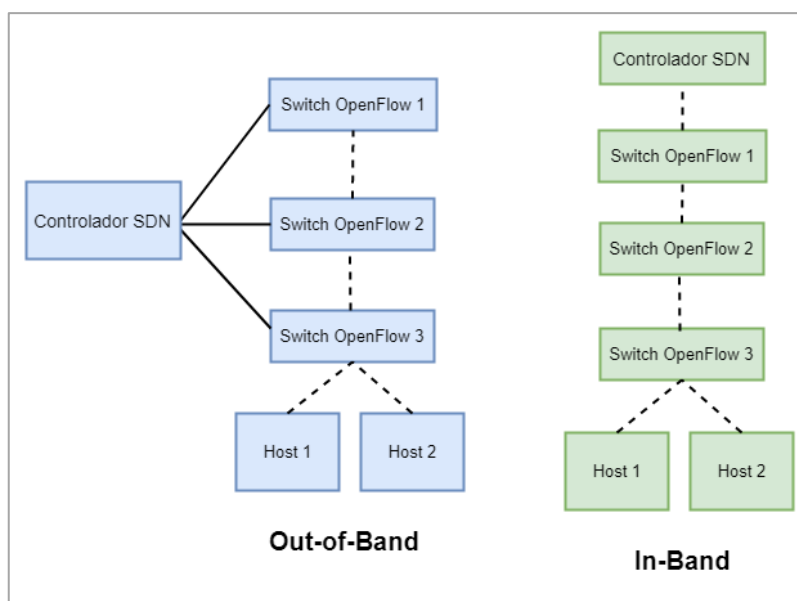
De acordo com Mckeown et al. (2008), "o papel do controlador é receber todo o tráfego enviado pelo *switch* e decidir a ação a ser aplicada ao tráfego recebido", fornecendo uma visão global do estado da topologia e o estado da rede.

Em relação a interação entre o controlador e os comutadores *OpenFlow*, existem duas formas de implementação, uma denominada *Out-of-Band*, onde a comunicação entre os comutadores *OpenFlow* e o controlador SDN ocorre por um canal diferente do utilizado para a

comunicação dos comutadores entre si. A outra forma é definida como *In-Band*, onde o caminho de comunicação entre o controlador SDN e os comutadores *OpenFlow* é o mesmo utilizado para a comunicação entre os comutadores (BENTON; CAMP; SMALL, 2013).

A Figura 5 ilustra a diferença entre esses tipos de implementação (*Out-of-Band* e *In-Band*) do Controlador dentro da rede.

Figura 5 – Implementação Out-of-Band e In-Band



Fonte: Benton, Camp e Small (2013).

É importante destacar que "embora o controlador seja logicamente centralizado" (SPALLA et al., 2015), ele pode ser implementado fisicamente de forma distribuída para garantir alta disponibilidade da rede (HALTERMAN, 2018).

2.4 SEGURANÇA EM REDES DE COMPUTADORES

A segurança é um dos aspectos mais relevantes no contexto das redes de computadores, uma vez que a tecnologia moderna, principalmente as redes computacionais, que estão inseridas em todos os níveis sociais no contexto atual, tornaram-se *commodities* e transparentes para seus usuários. É indiscutível a importância da *Internet* nos dias atuais e, conseqüentemente, das redes de computadores que a apoiam (COSTA; PEREIRA; ARA, 2012).

Com a disseminação das redes de computadores e o advento e evolução da *Internet*, a segurança tornou-se uma das maiores preocupações para seus usuários. A estrutura complexa da *Internet* assim como suas facilidades, contribuíram com o surgimento de diversas ameaças

de segurança no contexto das redes de computador (GAIGOLE; KAMALTAI; KALYANKAR, 2015).

Outro ponto-chave a ser observado é o nível de sofisticação das ferramentas para capturar o tráfego, capturar senhas, “quebrar” criptografia e explorar vulnerabilidades de naturezas diversas (MORIMOTO, 2011).

Os danos causados por "códigos maliciosos, hackers e ataques de negação de serviço, estão se tornando cada vez mais comum, mais ambicioso e incrivelmente sofisticado" (ABNT, 2002).

Tais preocupações com os aspectos de segurança são necessárias e extremamente importantes, pois existem grandes quantidades de informações sensíveis (dados pessoais, comerciais, militares e governamentais) na infraestrutura das redes em todo o mundo (GAIGOLE; KAMALTAI; KALYANKAR, 2015).

Nos dias atuais, é possível verificar o crescente número de vulnerabilidades em sistemas de computador e nas tecnologias de rede. Este fato influencia diretamente no aumento dos ataques nestes sistemas (PINHEIRO, 2007).

2.4.1 Principais Ameaças em Redes de Computadores

A tecnologia introduziu muitas vulnerabilidades e riscos para os usuários em geral. As redes de computadores viabilizaram e intensificaram crimes que anteriormente eram inexistentes, rudimentares e de baixo impacto.

Nesse contexto, os atacantes exploram as vulnerabilidades dos sistemas informáticos, principalmente das redes computacionais, a fim de realizar diversos tipos de ataques, com diferentes objetivos (LOPEZ, 2014).

Existem inúmeros métodos de ataque e ferramentas para exploração de vulnerabilidades em sistemas de redes de computador. Alguns dos principais métodos encontrados na literatura são: varredura em redes (*Scan*), interceptação de tráfego (*Sniffing*), força bruta (*Brute Force*), negação de serviço (DoS e DDoS), falsificação e homem no meio (*Man In The Middle*) (BASSO, 2010):

- a) **Varredura em Redes (*Scan*):** A técnica de varredura em redes ou *Scan*, consiste no uso de ferramentas para realizar buscas detalhadas em redes de computadores, visando identificar dispositivos ativos e coletar informações de diferentes naturezas. Esse tipo de ataque pode ser usado maliciosamente, a fim de descobrir vulnerabilidades e capturar informações confidenciais em uma rede. Pode

igualmente ser usado de uma maneira benéfica por administradores de rede para a análise do comportamento do tráfego (CERT.BR, 2012);

- b) **Interceptação do Tráfego (*Sniffing*):** Os *sniffings* ou *sniffers* são *softwares* usados para inspecionar dados que viajam em uma rede de computadores. Essas ferramentas podem capturar todo o tráfego que passa em um segmento de rede. Utilizadas de forma maliciosa, essas ferramentas podem gerar quebra de confiabilidade e captura de senha (JUSTI, 2016);
- c) **Força bruta (*Brute Force*):** Basicamente, esta técnica consiste em adivinhar por tentativa e erro, nomes de usuário e senhas válidas para acessar um determinado sistema ou dispositivo. O invasor usa informações coletadas no contexto ou *scripts* chamados dicionários que armazenam um conjunto estratégico de palavras-chave e caracteres que podem ser usados por *softwares* maliciosos para aplicar a técnica de força bruta (CERT.BR, 2012; BASSO, 2010);
- d) **Negação de serviço (DoS e DDoS):** Ataques de negação de serviço não procuram invadir ou coletar informações, mas esgotar recursos e causar indisponibilidade de serviços, dispositivos ou toda a rede. Nos casos mais recorrentes, um invasor aciona inúmeras requisições para um dispositivo de destino em uma rede com o objetivo de executar mais solicitações do que podem ser atendidas. Quando o ataque é executado de um único computador, é chamado DoS (*Denial of Service*), porém quando executado em uma maneira distribuída usando diversos dispositivos, o ataque é denominado de DDoS (*Distributed Denial of Service*) (RIBEIRO et al., 2012; CERT.BR, 2012). Alguns dos ataques de negação de serviço mais recorrentes citados na literatura são: *SYN Flood*, que esgota os recursos de um servidor através de uma conexão TCP e *Smurf* que basicamente torna uma rede ou dispositivo de rede indisponível, através de pacotes de inundação com várias solicitações de mensagens do tipo ICMP (*Internet Control Message Protocol*) por meio de *pings* (CERT.BR, 2012);
- e) **Spoofing:** Nesta técnica, o invasor usa um computador para falsificar informações (personificar) de um determinado dispositivo ou usuário na rede, fazendo com que outro falso ocupe o lugar do original. Em uma rede de computadores este tipo de ataque envolve a personificação do endereço de origem. Por meio desse tipo de técnica, um invasor pode, por exemplo, falsificar seu endereço de rede usando um endereço confiável e bem conhecido na rede para enganar os dispositivos que fazem

- parte do domínio de rede (RIBEIRO et al., 2012; BASSO, 2010). Os ataques de *spooffing* podem ser dirigidos aos recursos de rede múltiplos tais como a falsificação do email (*Email Spooffing*), o serviço DNS (*DNS spooffing*), os endereços IP (*IP spooffing*) e os pacotes ARP (*ARP Spooffing* ou *ARP Poison*) (CERT.BR, 2012), e;
- f) **Man In The Middle:** Esta é uma técnica em que o atacante se posiciona entre uma transmissão válida com o objetivo de interceptar os dados da transmissão. De uma forma mais específica, um ataque de *Man In The Middle* ou o homem no meio, permite ao invasor a capacidade de ler, inserir e modificar mensagens trocadas entre dois dispositivos na rede, sem que as entidades saibam que a conexão entre eles está comprometida (SHUBH; SHARMA, 2016).

Em contraste com cenário de crescentes ameaças (*Scanners* de rede, *Sniffings*, DoS, DDoS, *Spooffing*, etc.) e o aumento da complexidade das redes e suas tecnologias, é inevitável a necessidade da implementação de procedimentos que busquem combater ameaças e controlar os riscos aos quais as informações são expostas.

2.4.2 Aspectos de Segurança em Redes – Principais Pilares

A segurança no contexto das redes de computadores se baseia em alguns princípios fundamentais, que são herdados dos conceitos gerais de segurança da informação. Pode-se dizer que a segurança de rede é uma das disciplinas dentro do universo da segurança da informação (ABNT, 2013b).

O foco da segurança em ambientes de rede computacional está relacionado com a proteção e garantias de confidencialidade, integridade e disponibilidade das informações (CERT.BR, 2012). Esses três princípios são os alicerces que apoiam todos os conceitos de segurança da informação e suas vertentes. No entanto, outros aspectos como a autenticidade, controle de acesso e a não-repúdio também são considerados para garantir uma maior eficiência na aplicação dos controles de segurança das políticas e das contramedidas (COUTINHO et al., 2017). Segue uma breve discussão sobre os principais alicerces ou pilares de segurança:

- a) **Confidencialidade:** Estabelece que todas as informações devem ser acessadas apenas por usuários devidamente autorizados (COUTINHO et al., 2017; CERT.BR, 2012).

- b) **Integridade:** Estabelece que toda e qualquer informação deve permanecer no mesmo estado em que foi disponibilizada e não pode ser alterada (acidentalmente ou intencionalmente) por usuários não autorizados. A integridade dos dados é uma característica indispensável nas redes computacionais, sendo implementada através de mecanismos de criptografia (COUTINHO et al., 2017; CERT.BR, 2012);
- c) **Disponibilidade:** Estabelece que toda e qualquer informação ou recurso disponibilizado, deve ser acessível sempre que solicitado pelos usuários legítimos. Em sistemas de rede, a disponibilidade é fundamental para o perfeito funcionamento dos serviços prestados, sendo geralmente aplicada através de estratégias de redundância de dispositivos (servidores, *switches*, roteadores, *links* e etc.) e técnicas de espelhamento (COUTINHO et al., 2017; CERT.BR, 2012);
- d) **Autenticidade:** Estabelece que cada usuário deve ser identificado a fim de confirmar a veracidade de sua identidade. A autenticidade é definida como um princípio que representa a capacidade de identificar de forma exata, os usuários de um determinado sistema protegido, confirmando sua identidade dentro da rede (COUTINHO et al., 2017; CERT.BR, 2012);
- e) **Controle de Acesso:** Basicamente, é uma limitação controlada do acesso a determinadas informações. Em redes de computadores, o controle de acesso garante a partir da autenticação do usuário que este tenha acesso a determinados níveis da rede para serviços, recursos e dados (COUTINHO et al., 2017; CERT.BR, 2012), e;
- f) **Não-repúdio:** Estabelece que qualquer ação realizada por um usuário deve ser registrada, para que tal usuário não possa negar as ações realizadas por ele em um determinado sistema. Em uma rede de computadores, esse princípio é implementado por meio de mecanismos de auditoria, monitoramento e *logs* de acesso (COUTINHO et al., 2017; CERT.BR, 2012).

2.4.3 Mecanismos de Segurança

Compreender os aspectos de segurança no contexto das redes computacionais é um requisito fundamental no processo de identificação e tratamento de ameaças recorrentes nesse tipo de ambiente.

No entanto, os benefícios da segurança da informação só são alcançados através da "implementação de um conjunto adequado de controles, incluindo políticas, processos, procedimentos, estrutura organizacional e funções de *software* e *hardware*" (ABNT, 2013b).

Com base nos princípios da segurança da informação e em todos os seus aspectos derivados, os mecanismos de segurança foram desenvolvidos e adaptados para ajudar a proteger as redes de computadores contra as ameaças físicas e lógicas, assegurando assim a confidencialidade, integridade, disponibilidade, autenticidade controle de acesso e não-repúdio (CERT.BR, 2012).

A ausência de tais mecanismos e procedimentos de segurança em ambientes de rede, possibilita o acesso indevido e a exploração de informações internas por usuários não autorizados (COSTA; PEREIRA; ARA, 2012).

Existem vários mecanismos e várias ferramentas de segurança especificadas e documentadas na literatura, no entanto, os principais mecanismos de segurança relacionados com as redes de computadores são: política de segurança da informação, autenticação, criptografia, cópias de segurança (*backups*), registro de eventos (*logs*), *antimalware*, *firewall* e sistemas IDS/IPS (MOLINA; SILVEIRA; SANTOS, 2015):

- a) **Políticas de Segurança da Informação:** No contexto das redes de computadores, uma PSI ou Política de Segurança da Informação é um documento que define as ações, técnicas, métodos, responsabilidades e direitos em relação ao acesso e ao uso adequado dos recursos disponíveis na rede, com o objetivo de proporcionar maior segurança. A PSI é a base para a implementação de todos os outros mecanismos de segurança (criptografia, *backup*, *logs* de eventos, etc.) Buscando maior eficiência na aplicação de metodologias de segurança (CERT.BR, 2012; ABNT, 2013b);
- b) **Autenticação:** é o mecanismo utilizado no processo de identificação de usuários dentro de sistemas e redes. Utiliza combinações de logins e senhas, tokens, certificados digitais, leitores biométricos, identificadores de íris ocular e entre outros métodos, relacionados ao processo de identificação e validação de usuários, com a finalidade de permitir ou negar acesso à rede e seus recursos (CERT.BR, 2012; ABNT, 2013b);
- c) **Criptografia:** Os mecanismos de criptografia destinam-se a disfarçar (*encriptar*) mensagens trocadas entre dispositivos de rede, ocultando o conteúdo original da mensagem de agentes não autorizados. O processo de criptografia e decriptografia é feito por algoritmos matemáticos baseados em função (CERT.BR, 2012; ABNT, 2013b(ABNT, 2013a));
- d) **Backups:** Os *backups* (cópias de segurança) garantem a preservação de informações de um sistema ou dispositivos em uma rede em caso de falhas graves, possibilitando a

- restauração dessas informações, evitando perdas totais ou parciais (CERT.BR, 2012; ABNT, 2013b(ABNT, 2013a));
- e) **Registros de Eventos (*logs*):** Os mecanismos de *log* possibilitam o registro das atividades que ocorrem em um determinado sistema ou rede. Eles podem ser armazenados em arquivos, na memória de dispositivos ou em bancos de dados. Os *logs* são usados para análise de uso indevido de recursos ou dispositivos de rede, detecção de ataques e ações de rastreamento (auditoria) realizadas em sistemas e dispositivos (CERT.BR, 2012; ABNT, 2013b);
 - f) **Software Antimalwares:** Os *softwares antimalware* são ferramentas desenvolvidas e implementadas com o objetivo de proteger, detectar, analisar ou remover códigos maliciosos em um dispositivo computacional. Atualmente o antivírus é a ferramenta mais utilizada nesta categoria, no entanto o antivírus fornece apenas proteção básica e em ambientes de rede não substituem o uso de soluções mais sofisticadas como *firewall* e sistemas de IDS/IPS (CERT.BR, 2012; ABNT, 2013b);
 - g) **Firewall:** Os *firewalls* são mecanismos projetados para proteger informações da rede privada. Basicamente, é um conjunto de *hardware* e *software* posicionados entre a rede privada e redes externas, como a *Internet*, e tem como objetivo principal isolar a rede privada de acesso não autorizado. Um *firewall* restringe a comunicação, tornando possível controlar, autenticar e auditar o tráfego de uma rede. É um mecanismo fundamental para evitar o tráfego não autorizado dentro de uma rede de computador (CERT.BR, 2012; ABNT, 2013b);
 - h) **IDS:** Os Sistemas de Detecção de Intrusão (*Intrusion Detection Systems – IDS*), são consideradas como principais ferramentas para análise passiva e ativa em tempo real de tráfego, uma vez que possibilitam a identificação de comportamento suspeito no tráfego de rede (CERT.BR, 2012; ABNT, 2013b). A técnica utilizada pelos IDS no processo de detecção de intrusão se baseia na “coleta e processamento de todo o tráfego dos dados que transitam pela rede, comparando-os com assinaturas ou comportamentos conhecidos, afim de encontrar atividades maliciosas e acesso não autorizado aos recursos da rede” (MORAIS, 2011). Existem diversos tipos de IDS, que podem ser baseados em hardware ou software, em versões *open source* ou privadas. Esses sistemas podem ser classificados em três grupos de acordo com suas funcionalidades, que são: (i)Sistemas de intrusão baseado em rede (*Network Intrusion Detection System – NIDS*), (ii)baseados em *host* (*Host-based Intrusion Detection – HID*) e (iii)sistemas

híbridos ou distribuídos (*Hybrid Intrusion Detection*). Os NIDS são sistemas que capturam e analisam todos os pacotes de um tráfego de rede à qual tem acesso, independente do meio de comunicação ou protocolo utilizado, com o objetivo de proteger os ativos de uma rede (BERALDO et al., 2018). Os HIDS tem como objetivo o monitoramento do tráfego direto de hosts isolados ou conjunto de hosts, possibilitando o gerenciamento das permissões e controle de acesso, emissão centralizada de alertas e diversos tipos de análises estatísticas e forense (BERALDO et al., 2018). Já os sistemas híbridos, possibilitam a junção dos métodos utilizados tanto nas abordagens baseadas em redes (NIDS) como das baseadas em hosts (HIDS), dessa forma o monitoramento ocorre tanto no tráfego geral da rede, como na *interface* dos dispositivos (*hosts*) da rede (CERTSI, 2017);

- i) **IPS:** Os Sistemas de Prevenção de Intrusão (*Intrusion Prevention System – IPS*) funcionam como um complemento do IDS, acrescentando a capacidade de reação a tentativas de intrusão baseadas em regras estabelecidas pelo administrador da rede (CAMELO et al., 2017). Os sistemas IPS são ferramentas de segurança ativas, que atuam de forma preventiva na rede, através de abordagens para identificação e bloqueio em tempo real de potenciais ameaças (NAKAMURA, 2016). Assim como as ferramentas IDS, os Sistemas de Prevenção de Intrusão podem ser classificados em grupos, os quais definem o modo de operação e implementação da ferramenta (CAMELO et al., 2017). Os sistemas baseados em rede NIPS (*Network Intrusion Prevention System*) que reagem a intrusões bloqueando ataques detectados na rede e os HIPS (*Host-based Intrusion Prevention System*) (VACCA, 2010). Os *IPS* são utilizados para conter diversos tipos de ameaças de rede, como ataques do tipo DDoS e *Man In The Middle* (CERTSI, 2017).

2.5 TRABALHOS RELACIONADOS – SEGURANÇA EM REDES SDN

É notório que as Redes Definidas por *Software* implementam novas possibilidades no contexto das redes de computadores. Várias vantagens, tais como escalabilidade, dinamismo, fácil gestão e controle de funções, programação e entre outras características facilitam a rápida inovação de recursos e tecnologias de rede (YOON et al., 2017; SHIN et al., 2014; PORRAS et al., 2012).

No entanto, além das inúmeras facilidades e benefícios, o paradigma SDN introduz novas vulnerabilidades e ameaças no contexto das redes de computadores (RÖPKE; HOLZ, 2015).

O trabalho apresentado em (PORRAS et al., 2012) introduz uma solução eficiente para uma vulnerabilidade crítica das redes SDN, que é o conflito entre diferentes regras de fluxo instaladas por aplicativos SDN. Esta vulnerabilidade permite que os aplicativos desenvolvidos por terceiros, consigam sobrepor as regras *OpenFlow* estabelecidas pelos administradores da rede, permitindo a mudança de regras relacionadas com as políticas de segurança estabelecidas na rede. Com o objetivo de resolver essa vulnerabilidade, é proposta uma solução denominada FortNox, desenvolvido para a implementação eficiente de políticas de segurança em ambientes de rede SDN. Basicamente, o FortNox é implementado como uma extensão do controlador NOX, que detecta e monitora em tempo real a inserção de regras conflitantes e trata essa vulnerabilidade atribuindo níveis de prioridades e assinaturas digitais às regras do *OpenFlow*, onde cada regra de prioridade mais alta não pode ser sobreposta por regras com prioridade mais baixa.

Uma análise conceitual das vulnerabilidades no contexto das redes SDN é apresentada em (ZERKANE et al., 2017), visando estudar suas fraquezas e medir os impactos das ameaças às quais estão expostas. A análise é feita levando em consideração cada uma das camadas existentes no contexto das redes SDN, mantendo o foco nas principais características como o controlador, o protocolo *OpenFlow*, os dispositivos de comutação e suas tabelas de fluxo na camada de encaminhamento de dados. O sistema comum de Pontuação de vulnerabilidade CVSS é usado (*Common Vulnerability Scoring System*) (FIRST, 2019) para uma análise do impacto das vulnerabilidades em redes SDN apresentadas no trabalho.

Uma avaliação dos aspectos vulneráveis do protocolo *OpenFlow* e a forma como são implementados pelos fornecedores de *hardware* é abordado em (BENTON; CAMP; SMALL, 2013). Os autores abordam as principais falhas na especificação e implementação do TLS pelos principais fornecedores de *hardware* e *software* baseados no protocolo *OpenFlow*. Em segundo lugar, os impactos (falta de confidencialidade, integridade e disponibilidade) de tais vulnerabilidades na comunicação entre o controlador (camada de controle) e os dispositivos de comutação (camada de dados) são analisados.

Em (LI; HONG; BOWMAN, 2011) é apresentado o estudo da arquitetura de segurança e a análise de vulnerabilidades no ambiente global para experimentação e inovação de redes computacionais denominadas ProtoGENAI. Os autores realizaram e documentaram uma variedade de experimentos visando identificar as vulnerabilidades no contexto do ProtoGENAI. Ataques direcionados para estudo (*sniffing* de pacotes, *spoofing* de pacotes, *bugs* de

virtualização e etc.) foram classificadas levando em consideração os três planos da arquitetura SDN (plano de aplicação, plano de controle e plano de dados).

Em (HAYWARD, 2015) os principais requisitos de segurança em projetos para o desenvolvimento dos controladores SDN são identificados. Os principais *softwares* de controle SDN de código aberto (OpenDaylight, ONOS e Ryu) foram analisados em relação aos aspectos de segurança. Aspectos como desempenho, resiliência, conflitos de políticas, canal de comunicação seguro com TLS, serviços de auditoria, autenticação e autorização são levados em consideração, e posteriormente são feitas recomendações indicando melhorias de segurança com o objetivo de aprimorar os Controladores SDN.

A análise de vulnerabilidades em controladores SDN é discutida em (YOON et al., 2017). As vulnerabilidades estudadas são classificadas considerando os principais aspectos da segurança da informação (confidencialidade, integridade e disponibilidade). Para avaliar o impacto real destas vulnerabilidades, os autores replicaram alguns dos principais ataques em um *testbed* físico, onde avaliaram três dos principais controladores SDN, FloodLight (PORRAS et al., 2015), OpenDayLight (BADOTRA; SINGH, 2017) e POX (GUEDES et al., 2012) e suas principais aplicações de segurança. Finalmente, os autores propõem algumas diretrizes para melhorar a segurança na implementação dos Controladores SDN e no desenvolvimento de aplicações de rede.

Em (MIGAULT; POURZANDI, 2015) é apresentada uma visão geral da arquitetura das redes SDN, assim como análises dos aspectos de segurança mais relevantes e as principais vulnerabilidades existentes neste contexto. Os autores levantam questões sobre autenticação e autorização, alta disponibilidade no plano de controle, auditoria, entre outros. Além das análises realizadas, são discutidas contramedidas com o objetivo de reduzir as vulnerabilidades existentes nas redes SDN.

A pesquisa apresentada em (KLOTI; KOTRONIS; SMITH, 2013) aborda um estudo sobre o protocolo *OpenFlow*, considerando as suas principais vulnerabilidades e os aspectos de segurança relevantes no contexto das redes SDN. Testes de vulnerabilidade foram executados em um *testbed* emulado com mininet (KAUR; SINGH; GHUMMAN, 2014) e controlador POX (GUEDES et al., 2012), com o objetivo de avaliar de forma eficientemente as ameaças em ambientes de rede SDN. A modelagem e análise de vulnerabilidades é feita usando a metodologia da *Microsoft* conhecida como STRID (*Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service e Elevation of Privilege*) (KHAN, 2017). Em segundo momento, é realizado ataques de *Man In The Middle* para explorar as vulnerabilidades

identificadas. Por fim, são feitas recomendações de segurança com base nas análises e testes realizados ao longo do trabalho.

Um conjunto de princípios básicos de segurança é apresentado em (ONF, 2016). O documento fornece vários critérios e instruções com o objetivo de projetar e implementar redes SDN mais seguras. Uma análise de segurança do protocolo *OpenFlow* e os *switches* que apoiam este protocolo são abordados em um segundo momento. As principais vulnerabilidades do protocolo *OpenFlow* e das redes SDN foram mapeadas. Finalmente, são sugeridas várias possibilidades para correção, visando orientar a implementação segura de Redes Definidas por Software.

As principais vulnerabilidades dos sistemas operacionais de rede (*Network Operation System* - NOS) ou *softwares* controladores, são objeto de estudo em (SHIN et al., 2014). Alguns aspectos são abordados, como aplicativos *OpenFlow* mal construídos ou desenvolvidos de forma maliciosa e como esses aplicativos vulneráveis podem bloquear o plano de dados e conseqüentemente, toda a rede SDN. Alguns testes e análise de vulnerabilidades foram feitos em alguns sistemas operacionais de redes SDN como: POX, NOX, Beacon (ERICKSON, 2013) e Floodlight. Em um segundo momento, o sistema operacional de rede Rosemary é proposto com o objetivo de implementar maior segurança no plano de controle, orientado à correção de problemas com aplicações maliciosas e falhas de projeto de sistema através de assinaturas digitais e isolamento de aplicativos com ferramentas de *sandbox*.

Em (MENEZES et al., 2014) é proposto o AuthFlow, um mecanismo de autenticação e controle de acesso de estações finais em Redes Definidas por *Software* que se baseiam em verificação de credenciais. Um protótipo é implementado no controlador POX para estudos e testes. Posteriormente são apresentados os resultados obtidos com o experimento e os testes realizados com a ferramenta AuthFlow.

Alguns experimentos conduzidos com o objetivo de analisar vulnerabilidades em redes SDN são propostos em (RÖPKE; HOLZ, 2015). Basicamente um *rootkit* (software malicioso para controle de sistemas) é instalado no controlador SDN. Através do *rootkit* os autores demonstram as possibilidades de exploração de vulnerabilidades tais como: acesso remoto não autorizado ao Controlador SDN, modificação de serviços de rede, inserção de regras maliciosas, mudança de regras de segurança, ocultação do estado real da rede, falsificação de estatísticas de rede, entre outros. Os protótipos são implementados e os testes realizados nos Controladores OpenDaylight e HP Controller.

Uma análise abrangente dos desafios de segurança relacionados ao contexto das redes SDN é apresentado em (HAYWARD; O'CALLAGHAN; SEZER, 2013). As vulnerabilidades detectadas são categorizadas de acordo com as camadas da arquitetura SDN (aplicativo, controle e dados). Posteriormente, são discutidas soluções propostas na literatura em contraste aos desafios analisados.

Em (SHU et al., 2016) é apresentado em um primeiro momento, a arquitetura das redes SDN de modo geral. Posteriormente, uma comparação das vantagens e desvantagens das redes SDN é feita em relação às redes tradicionais. Em um segundo momento, um estudo analítico é realizado com foco nas principais vulnerabilidades e ameaças da arquitetura SDN e posteriormente são propostas algumas contramedidas de segurança.

Uma proposta de contenção de ataques DDoS é apresentada em (CORRÊA et al., 2016). A técnica chamada SHADE (*Selective High Rate DDos Defense*) é proposta com o objetivo de aumentar a disponibilidade do plano de controle na arquitetura SDN. Toda a arquitetura operacional da técnica e suas principais características são apresentadas, bem como alguns gráficos de teste que demonstram a eficiência da ferramenta na contenção de ataques de negação de serviço.

A ferramenta VeriFlow é proposta em (KHURSHID et al., 2012) como uma solução para vulnerabilidades relacionadas a conflitos e sobreposição de regras *OpenFlow*, através do monitoramento da mudança no estado de rede em tempo real, fornecendo regras de aplicação *OpenFlow* de forma mais eficiente e segura. Por fim, são apresentados os resultados dos testes realizados com a ferramenta, levando em conta o tempo de resposta, desempenho, latência, entre outros aspectos.

Os aspectos de segurança na implementação da arquitetura SDN em cenários de *Data Center* são estudados em (ONF, 2013). Um experimento relacionado à quarentena de *malware* de forma automatizada, é apresentado para ilustrar como a arquitetura SDN pode melhorar a segurança em ambiente de rede, isolando automaticamente dispositivos comprometidos. Os benefícios e as melhores práticas são apresentados posteriormente.

2.6 CONCLUSÃO

Nesse capítulo foram apresentados alguns dos principais conceitos relacionados à segurança em redes de computadores abordados na literatura e que serviram como suporte para o desenvolvimento desse trabalho.

Foram abordados os principais aspectos da Qualidade de Serviço, Qualidade de Experiência, Qualidade de Dispositivos, abordagens sobre contexto, arquitetura das redes SDN e as principais características da segurança da informação aplicadas às redes de computadores para resolver problemas recorrentes das tecnologias de rede legadas, assim como problemas emergentes com os novos paradigmas e tecnologias híbridas .

Alguns trabalhos correlatos a essa pesquisa também foram apresentados e serviram como orientação e comparativo para o desenvolvimento das propostas e soluções dessa dissertação de mestrado.

3 CAARF-SDN: ARCABOUÇO BASEADO EM CONTEXTO

3.1 INTRODUÇÃO

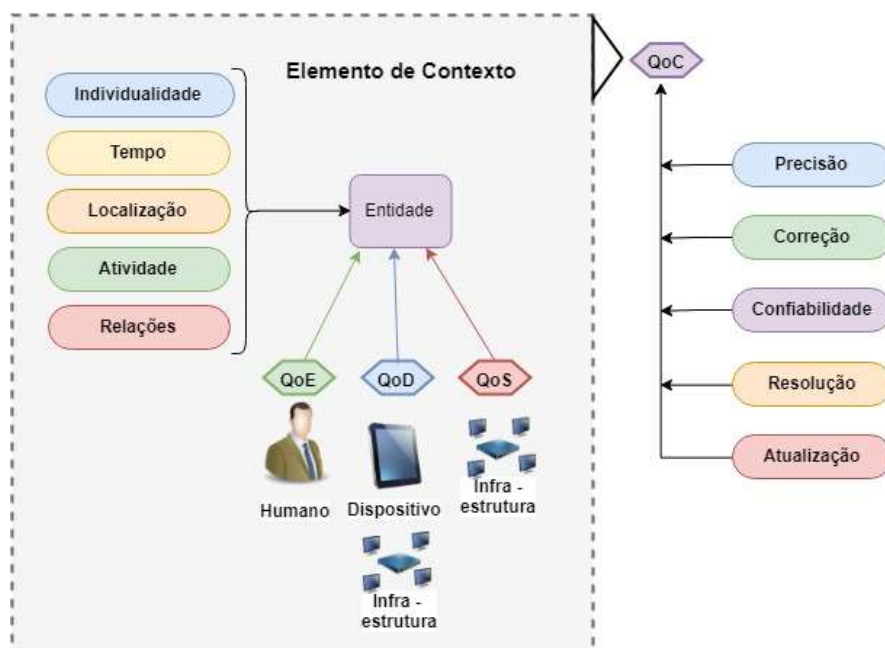
Os principais conceitos e princípios da arquitetura do CAARF-SDN são apresentados neste capítulo, buscando elucidar as principais diferenças e evoluções em relação ao projeto original do arcabouço *Context-Aware Adaptive Routing Framework* (CAARF). São detalhadas suas principais funcionalidades e cada um dos diferentes módulos que compõem sua arquitetura, assim como sua integração e interação com as redes SDN.

3.2 MODELO DE CONTEXTO

Segundo Badaró Neto et al. (2018) o modelo de contexto utilizado no projeto CAARF-SDN foi apresentado e discutido em (OLIVEIRA et al., 2015) o qual introduz as notificações baseadas em contexto na versão inicial do projeto CAARF. O modelo consiste na abordagem de um sistema computacional, que é descrito através da junção de conceitos como a experiência do usuário (Qualidade de Experiência – QoE), dos dispositivos de apresentação e de rede (Qualidade de Dispositivo – QoD) e pelas condições de tráfego (Qualidade de Serviço – QoS) (BADARÓ NETO et al., 2018).

A Figura 6 ilustra o modelo Conceitual desenvolvido para o projeto original do CAARF e posteriormente utilizado no CAARF-SDN.

Figura 6 – Modelo de Contexto do CAARF



Fonte: Oliveira et al. (2015).

Com o apresentado em (OLIVEIRA et al., 2015), através da categorização das variáveis QoS, QoD e QoE, o contexto pode ser descrito “e associado com parâmetros de qualidade” abordados na literatura. Dessa forma é possível estabelecer a relevância de uma entidade de forma exclusiva, levando em consideração que os valores aplicados nas políticas do modelo contextual devem ser precisos, passíveis de correção, confiáveis e atuais (MUAKAD, 2015).

Por fim, o conjunto resultante das informações descritas nos parâmetros de QoS, QoD e QoE é denominado de *Qualidade de Contexto* (QoC), o qual representa o nível de visão global da qualidade percebida pelo o usuário dentro desse modelo (SPINOLA, 2015).

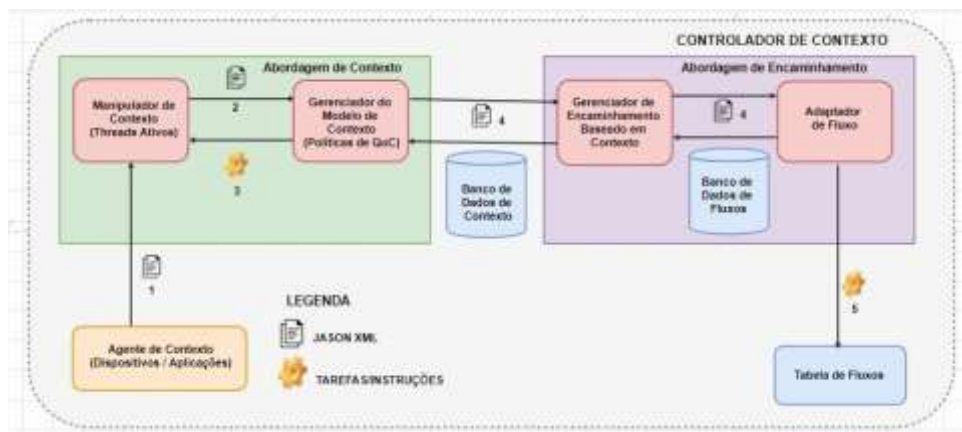
Na próxima seção, a arquitetura, principais características e funcionamento do arcabouço CAARF e CAARF-SDN são apresentadas detalhadamente.

3.3 ARQUITETURA DO CAARF-SDN

Antes de apresentar o funcionamento e as principais características da arquitetura do CAARF-SDN, é necessária uma introdução conceitual sobre os principais aspectos do projeto inicial do arcabouço.

De modo geral, o arcabouço CAARF (MUAKAD, 2015) é uma aplicação que tem por objetivo final a otimização e utilização de recursos de rede e a entrega de tráfego baseado em contexto, ou seja, baseado no estado do sistema computacional (QoE, QoS e QoD), e em particular no QoC (BADARÓ NETO et al., 2018). A primeira versão da arquitetura conceitual do CAARF é ilustrada na Figura 7.

Figura 7 – Controlador de Contexto Projeto CAARF



Fonte: Oliveira (2015b).

Como apresentado na Figura 3, o componente “Controlador de Contexto” (*Context Controller*) é descrito como modelo conceitual do projeto CAARF e está subdividido em 2 módulos: “Abordagem de Contexto” (*Context Approach*) e “Abordagem de Encaminhamento” (*Forwarding Approach*) (OLIVEIRA, 2015b; SPINOLA, 2015).

- a) **Módulo de Abordagem de Contexto:** Esse módulo é subdividido em duas partes: Manipulador de Contexto (*Context Handler*) e o Gerenciador do Modelo de Contexto (*Context Model Management*). Nesse cenário, o Manipulador de Contexto interage com o ambiente, recebendo notificações de QoE, QoD e QoS através dos agentes de contexto que monitoram os dispositivos de rede verificando aspectos de integridade das informações e posteriormente armazenando na base de dados estabelecida para informações contextuais (OLIVEIRA, 2015b; SPINOLA, 2015). Como discutido em (BADARÓ NETO et al., 2018), o Manipulador de Contexto é responsável ainda por notificar o Gerenciador de Modelo de Contexto, que por sua vez faz análises das informações recebidas aplicando as políticas de QoC de acordo com a necessidade (MUAKAD, 2015).
- b) **Módulo Abordagem de Encaminhamento:** Esse módulo é responsável por avaliar e determinar possíveis rotas ou melhores caminhos através da execução do algoritmo de encaminhamento presente no sub-módulo Encaminhamento Baseado em Contexto (*Context-based Forwarding Management*) (MUAKAD, 2015; OLIVEIRA, 2015b). O segundo componente desse sub-módulo é o Adaptador de Fluxo (*Flow Adaptation*), que faz a leitura das informações processadas e reconfigura a rede aplicando novas regras de acordo com as mudanças necessárias no contexto da rede, seja ela convencional ou baseada no paradigma SDN (SPINOLA, 2015).

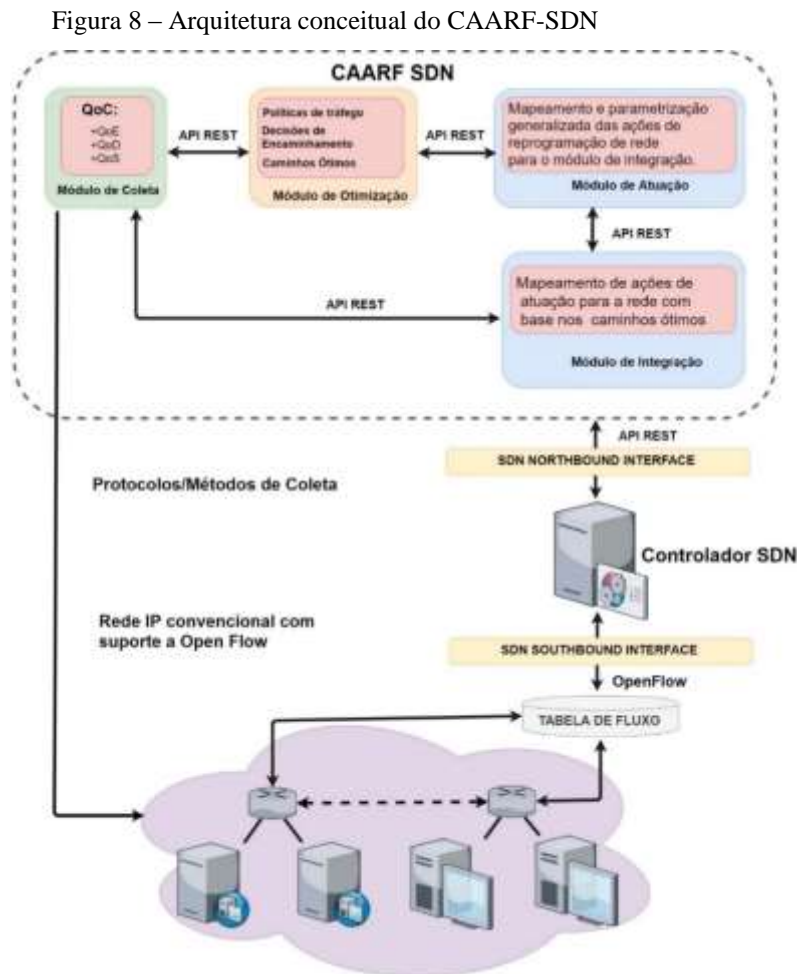
De acordo com Badaró Neto et al. (2018) o arcabouço CAARF-SDN é um serviço de rede derivado do projeto original do CAARF que atua na otimização e configuração dinâmica de uma rede SDN, acrescentando a leitura e armazenamento de informações contextuais.

Toda interação entre o CAARF e o ambiente de rede SDN é feita através da interface *Northbound*, que habilita a comunicação do controlador SDN com aplicações externas (BADARÓ NETO et al., 2018).

Ao propor a integração do arcabouço CAARF com paradigma SDN, diversas mudanças se fizeram necessárias para promover o funcionamento adequado do arcabouço e todos os seus

recursos, entretanto apesar das mudanças efetuadas na estrutura do projeto CAARF, o modelo resultante global do projeto se manteve similar.

Segundo Badaró Neto et al. (2018), o CAARF-SDN é composto por quatro módulos, com responsabilidades bem definidas: Coleta, Otimização, Adaptador de Fluxo, que foi dividido em *Atuação e Integração* na nova arquitetura proposta pelo CAARF-SDN. A Figura 8 apresenta uma visão conceitual da arquitetura do CAARF-SDN.



Fonte: Badaró Neto et al. (2018).

Segundo apresentado em (OLIVEIRA, 2015b), a arquitetura do arcabouço CAARF-SDN foi projetada visando um alto nível de independência entre os módulos, tornando-os heterogêneos, o que viabiliza sua interação e execução em diferentes arquiteturas e diversas tecnologias, tendo como único requisito específico, o suporte ao protocolo HTTP (*Hipertext Transfer Protocol*), visando garantir a interoperabilidade entre as aplicações (SPINOLA, 2015).

A independência dos módulos do CAARF-SDN, não afetam o desempenho do sistema no processo de reconfiguração dinâmica da rede, uma vez que o tempo de resposta do arcabouço é relativamente bom (OLIVEIRA, 2015b).

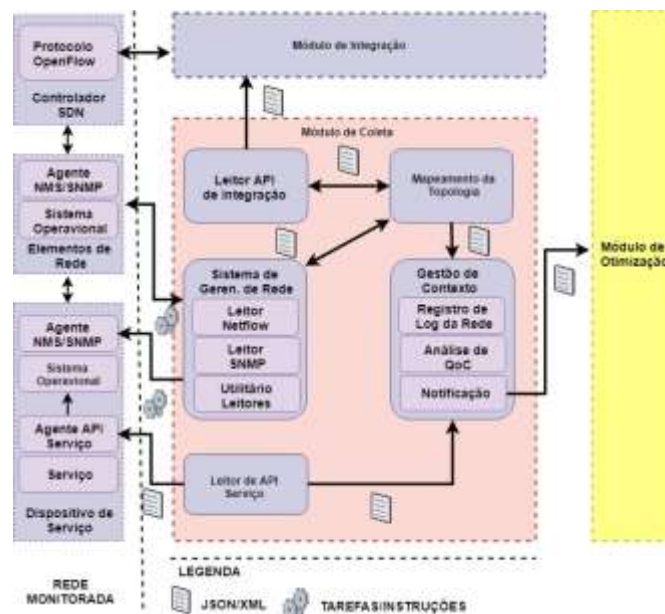
Nas sessões que seguem, são detalhados os principais componentes do arcabouço CAARF-SDN, assim como suas funcionalidades e a sua abordagem operacional.

3.3.1 Módulo de Coleta

Esse módulo é responsável por monitorar e analisar todo o estado da rede através da captura de dados referentes aos parâmetros QoS, QoD e inferindo estimativas sobre o QoE. O processo de monitoração e análise é executado de forma contínua buscando identificar variações no estado global do sistema (QoC) e enviando notificações de contexto para o Módulo de Otimização sempre que mudanças significativas são detectadas. Outra tarefa fundamental desse módulo é o papel de determinar o estado global do sistema computacional (através do cálculo de QoC) (BADARÓ NETO et al., 2018).

A Figura 9 ilustra de forma conceitual a arquitetura do módulo de coleta, apresentando suas interações e seu funcionamento de forma geral.

Figura 9 – Modelo conceitual do Módulo de Coleta do CAARF-SDN



Fonte: Badaró Neto et al. (2018).

Dentro do Módulo de Coleta ocorre a interação constante entre diversos sub módulos, responsáveis por realizar a integração com o Módulo de Otimização, mapear a topologia da

rede, efetuar análise de logs, gerenciar a rede e seus serviços e fazer a gestão de todas as informações do contexto da rede (BADARÓ NETO et al., 2018).

De acordo com Badaró Neto et al. (2018), o Módulo de Coleta monitora e analisa constantemente o estado global da rede, e ao detectar qualquer alteração no QoC dos dispositivos da rede que resulte em um valor menor que 2 (dois), uma notificação de contexto (notificação do tipo JSON) é gerada e enviada para o Módulo de Otimização.

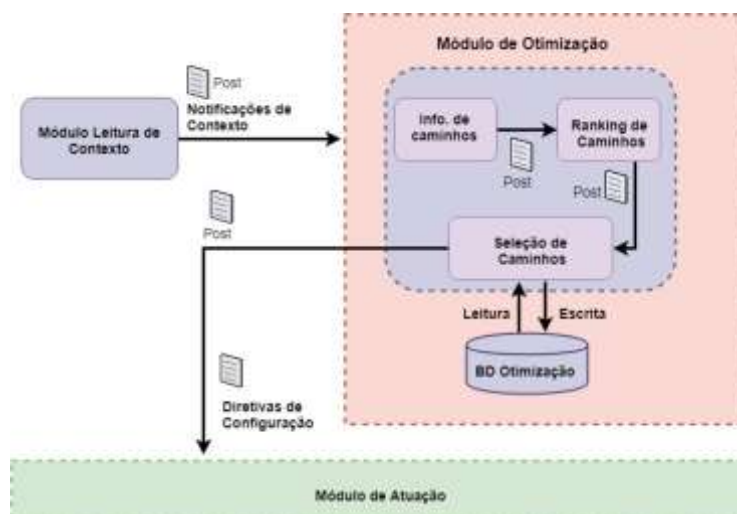
3.3.2 Módulo de Otimização

O módulo de Otimização é responsável por analisar as notificações de contexto enviadas pelo Módulo de Coleta e avaliar se mudanças devem ser efetuadas no encaminhamento dos dados. De forma mais clara, é esse módulo que determina o melhor caminho através da análise das melhores opções de encaminhamento. Para Badaró Neto et al. (2018), o Módulo de Otimização tem um papel fundamental na aplicação de decisões para melhorar a utilização dos recursos da rede, garantindo o melhor Acordo de Nível de Serviço (*Service Level Agreement – SLA*) entre os serviços ofertados e as necessidades dos usuários.

Os melhores caminhos podem ser identificados calculando a métrica de todos os fluxos, uma vez que o Módulo de Otimização recebe do Módulo de Coleta, informações cruciais sobre toda a topologia da rede monitorada, assim como cada custo de QoC dos elementos de rede pertencentes ao domínio (BADARÓ NETO et al., 2018).

Dentre todos os módulos do CAARF-SDN, esse é o mais “independente em relação às tecnologias adotadas”, uma vez que envia e recebe instruções via API REST dentro de uma especificação própria (BADARÓ NETO et al., 2018). Essa característica independente, torna o Módulo de Otimização versátil e aplicável em qualquer ambiente de rede independente da tecnologia utilizada. O grande diferencial dessa solução em relação a outras abordagens de engenharia de tráfego é a capacidade de trabalhar com diversos valores de QoC para calcular o custo total de cada caminho de rede. As soluções convencionais consideram parâmetros triviais como QoS, quantidade de saltos (distância) ou uma mescla de soluções (BADARÓ NETO et al., 2018). A Figura 10 ilustra a arquitetura do Módulo de Otimização de forma conceitual.

Figura 10 – Modelo conceitual do Módulo de Otimização do CAARF-SDN



Fonte: Badaró Neto et al. (2018).

Ao receber as informações do ambiente de rede e suas variações (QoE, QoD e QoS => QoC) denominadas de Notificações de Contexto, o Módulo de Otimização processa as informações e efetua a segregação relacionadas a caminhos, posteriormente encaminha os dados resultantes para o “Processador de Caminhos Baseados em Contexto” (BADARÓ NETO et al., 2018). Após o processamento das informações de caminhos, uma avaliação dos resultados é efetuada, com o objetivo de verificar ocorrências de violação de QoC notificadas que possam gerar alguma “Alteração da Política” de encaminhamento em vigor na rede (BADARÓ NETO et al., 2018). De acordo com badaró Neto et al. (2018), no caso da violação de QoC, é acionado o algoritmo de otimização que realiza cálculos de custo total de cada caminho de rede, utilizando diversos valores de QoC das portas dos dispositivos de encaminhamento, gerando alteração nas políticas de encaminhamento.

Com base nos resultados obtidos nos cálculos realizados pelo processo, é efetuada uma nova classificação (*ranking*) e a rede é reconfigurada de acordo com as novas políticas estabelecidas, gerando instruções (chamadas de diretivas de configuração) para reconfigurar as tabelas de fluxo, buscando adequar a rede SDN às mudanças do contexto (BADARÓ NETO et al., 2018).

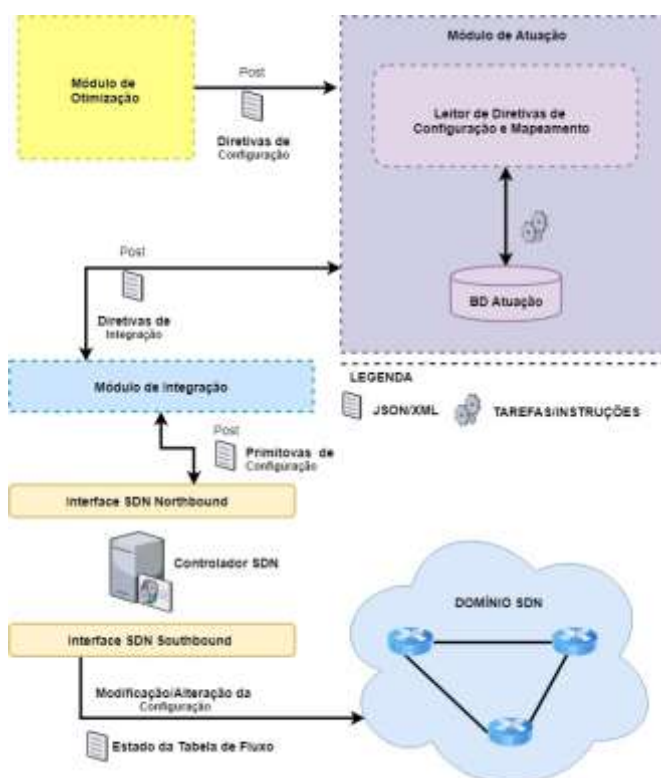
Ao final dos processos do Módulo de Otimização, todas as informações resultantes são enviadas para o Módulo de Atuação, responsável por aplicar as novas regras estabelecidas para a rede, com apoio do Módulo de Integração (BADARÓ NETO et al., 2018).

3.3.3 Módulo de Atuação

O Módulo de Atuação tem a função de reconfigurar as tabelas de fluxo da rede SDN, através da aplicação das instruções recebidas do Módulo de Otimização. Essas instruções são descritas por meio de uma API REST que abstrai os detalhes de implementação do software utilizado pelo Controlador SDN, característica que torna o CAARF-SDN totalmente independente do Sistema Operacional de Rede (*Network Operation System - NOS*) (BADARÓ NETO et al., 2018).

A Figura 11 apresenta o modelo conceitual da arquitetura do Módulo de Atuação demonstrando a interação entre seus processos.

Figura 11 – Modelo conceitual do Módulo de Atuação do CAARF-SDN



Fonte: Badaró Neto et al. (2018).

De acordo com Badaró Neto et al. (2018), o Módulo de Atuação pode ser dividido quatro processos principais. O primeiro é representado pelas instruções de reconfiguração (**Diretivas de Configuração**) das tabelas de fluxo geradas no Módulo de Otimização, representadas em um arquivo JSON. O segundo é representado pelo submódulo “Leitor de Diretivas”, responsável por realizar a identificação das tabelas de fluxos, rotas, elementos de rede e suas respectivas portas. O terceiro processo do “Módulo de Atuação” é o mapeamento, onde ocorre

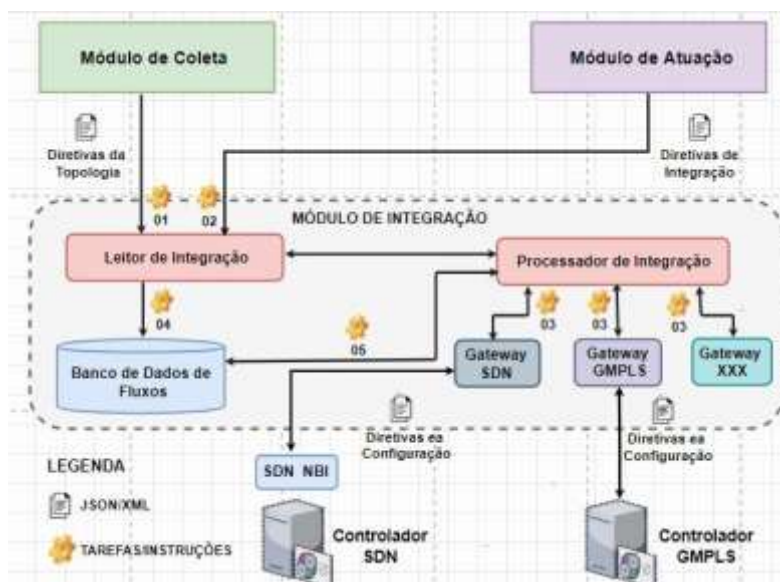
a análise das informações recebidas e a comparação entre a nova configuração de topologia sugerida e a configuração da rede em vigor. Por fim, o quarto processo é a geração de arquivos (JSON) de notificação, com instruções de alto nível denominadas de “Primitivas de Configuração”. O objetivo dessas diretivas é instruir adequadamente o Módulo de Integração no que diz respeito às alterações que devem ser realizadas no controlador SDN (BADARÓ NETO et al., 2018).

Ao término de todas as etapas realizadas no Módulo de Atuação, as instruções necessárias para aplicação das novas configurações de rede são adequadamente enviadas para o Módulo de Integração, como foi descrito anteriormente.

3.3.4 Módulo de Integração

Para viabilizar a comunicação entre o CAARF-SDN e o controlador de uma Rede Definida por *Software*, é necessária uma interface comum entre eles. De acordo com Badaró Neto et al. (2018), o Módulo de Integração faz o papel de interface entre os Módulos de Coleta e de Atuação em relação ao Controlador SDN. Como já citado, a arquitetura do CAARF-SDN foi pensada dessa forma para dar autonomia ao seu protótipo em relação ao sistema operacional de Redes SDN utilizado. A Figura 12 apresenta a estrutura conceitual do Módulo de Integração.

Figura 12 – Arquitetura Conceitual do Módulo de Integração



Fonte: Badaró Neto et al. (2018).

Na arquitetura ilustrada na Figura 12, existem dois componentes principais: O “Leitor de Integração”, que possui uma API acessível aos Módulos de Coleta e Atuação, e o “Processador

de Integração” (BADARÓ NETO et al., 2018), componente responsável por traduzir as requisições feitas pelo Módulo de Atuação ao “Leitor de Integração”.

De acordo com Badaró Neto et al. (2018), o Módulo de Coleta solicita informações sobre a topologia da rede de duas formas. A primeira é por “Requisições Agendadas” (*request*) do tipo GET a cada “X” minutos, solicitando ao Módulo de Integração sobre a topologia da rede, onde “X” é um valor maior que 1 (um) e pode ser configurado (BADARÓ NETO et al., 2018). A segunda forma é através do “Disparo Via Alertas” e pode ser utilizado somente em situações onde os dispositivos da rede suportam o protocolo de coleta SNMP (*Simple Network Management Protocol*), e onde o Módulo de Coleta pode forçar a coleta da topologia da rede através do disparo de eventos. Esses alertas podem ser utilizados caso algum limite do sistema seja alcançado, sendo assim, o disparo via alertas pode diminuir o tempo da análise (BADARÓ NETO et al., 2018).

Já o Módulo de Atuação, envia instruções de reconfiguração da rede para o “Módulo de Integração”, que podem ser disparadas de duas formas (BADARÓ NETO et al., 2018). A primeira é através das “Diretivas de Integração”, que nada mais são do que mensagens do tipo *request* POST, com instruções de reconfiguração da rede, enviadas de forma automática pelo Módulo de Atuação (BADARÓ NETO et al., 2018). A segunda forma é através de “Instruções do Administrador da Rede”, onde o administrador da rede pode realizar ajustes de configurações de forma manual, através de mensagens do tipo *resquest* POST. Geralmente esse procedimento é comum no processo de implementação de novos ambientes de rede (BADARÓ NETO et al., 2018).

Uma vez em operação, é permitida uma interação transparente entre o serviço CAARF-SDN e o controlador SDN, possibilitando a automação de configuração das tabelas de fluxo baseadas nas mudanças do contexto da rede (BADARÓ NETO et al., 2018). Ao levar em consideração as diversas informações contextuais, o CAARF-SDN possibilita melhor experiência do usuário em relação aos recursos e serviços da rede como um todo.

Segundo Badaró Neto et al. (2018), para possibilitar integração de outras tecnologias com o CAARF-SDN, é fundamental o desenvolvimento de uma API específica (*Gateway*) para cada tipo de tecnologia de rede adotada, o que pode tornar o processo de integração uma tarefa complexa, uma vez que nem sempre uma API REST estará disponível por parte do controlador.

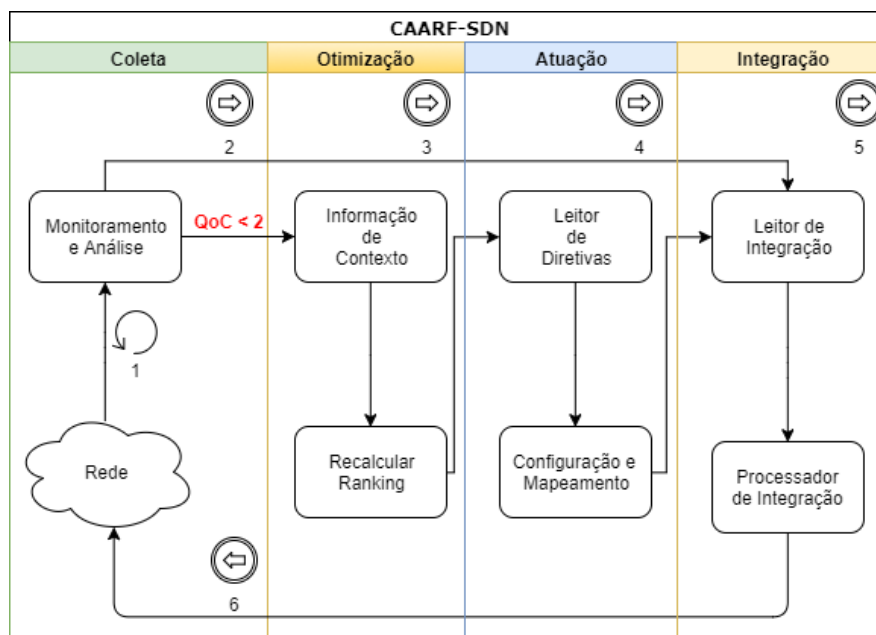
Na próxima seção é descrita a abordagem operacional do arcabouço CAARF-SDN, de uma forma mais generalizada.

3.4 GESTÃO DE ENCAMINHAMENTO DO CAARF-SDN

Esta seção apresenta o controle de fluxo completo do arcabouço CAARF-SDN, abstraindo os detalhes que foram mencionados nas seções anteriores.

A Figura 13 ilustra o fluxograma do ciclo operacional do arcabouço, apresentado como uma solução de roteamento dinâmico sensível ao contexto.

Figura 13 – Fluxograma da Abordagem Operacional do CAARF-SDN



Fonte: Badaró Neto et al. (2018).

Segundo Constantino (BADARÓ NETO et al., 2018), todo o processo operacional do CAARF-SDN é cíclico e contínuo, podendo ser resumido em 6 passos:

- No Módulo de Coleta, a captura de informações da rede é realizada de forma contínua, através de requisições relacionadas à topologia, seja por meio de captura de dados via SNMP, na rede convencional ou através de busca de informações no Módulo de Integração;
- A cada leitura do contexto é efetuado o recálculo de QoC. A cada novo cálculo, o valor de QoC resultante é comparado ao parâmetro de valor 2 (que pode ser configurável pelo administrador da rede). Caso o resultado do cálculo seja menor do que o parâmetro estabelecido, é gerada uma notificação para o Módulo de Otimização, dando início ao processo de reconfiguração dinâmica da rede;
- O algoritmo responsável por calcular os caminhos ótimos é executado pelo Módulo de Otimização, tomando como base, as informações referentes à topologia

e aos dados de QoC. Os dados resultantes são convertidos em diretivas de encaminhamento, que são enviadas ao Módulo de Atuação;

- d) O Módulo de Atuação é responsável por traduzir as diretrizes recebidas do Módulo de Otimização em primitivas de configuração que serão enviadas ao “Módulo de Integração”. É importante salientar que esse módulo não se comunica diretamente com o Controlador SDN, buscando evitar a dependência do arcabouço com tecnologias externas;
- e) A realização das possíveis alterações dos fluxos é efetuada no Módulo de Integração e posteriormente direciona as instruções para o *gateway* referente à tecnologia escolhida, e;
- f) Por fim, o *gateway* referente à tecnologia escolhida executa a reconfiguração das tabelas de fluxo com os novos parâmetros estabelecidos.

3.5 CONCLUSÃO

O projeto CAARF-SDN foi proposto com o objetivo de interagir com diversas tecnologias de rede em um ambiente híbrido, entretanto é fundamental para o funcionamento adequado da solução, que o paradigma SDN esteja em uso, uma vez que a capacidade de reprogramação dos elementos da rede é um fator crucial para viabilizar a implementação do arcabouço. A estratégia de separação da arquitetura em módulos, possibilita a interoperabilidade das tecnologias utilizadas, viabilizando a utilização de tecnologias diferentes na implementação de cada módulo, uma vez que a base de comunicação entre os módulos é a API REST.

O arcabouço CAARF-SDN propõe ambientes de redes híbridas mais eficientes e dinâmicos, viabilizando alto potencial de inovação e pesquisa para novas tecnologias de rede computacional que exigem maior escalabilidade e flexibilidade como é o caso da Internet das Coisas (*Internet of Things* - IoT), Computação na Nuvem (*Cloud Computing*), Big Data, entre outros.

Mesmo com todos os benefícios apresentados em relação à implementação do arcabouço CAARF-SDN, aspectos como segurança em ambientes de redes devem ser levados em consideração, uma vez que o aumento da complexidade e a interação entre diferentes tecnologias em ambientes híbridos acarretam no aumento do riscos, das vulnerabilidades e das possibilidades de ataques virtuais ou físicos com objetivos de comprometer o funcionamento parcial ou total da rede e seus recursos.

O próximo capítulo desta dissertação apresenta um estudo aprofundado sobre os principais aspectos de segurança no contexto do projeto CAARF-SDN e do contexto das Redes Definidas por Software, destacando seus principais pontos vulneráveis e identificando os principais ataques relacionados.

4 ASPECTOS DE SEGURANÇA DA INFORMAÇÃO APLICADOS AO ARCABOUÇO CAARF-SDN

4.1 INTRODUÇÃO

Devido aos diversos fatores relacionados a vulnerabilidades e ameaças no contexto das redes de computadores, o processo de implementação do arcabouço CAARF-SDN deve levar em consideração os problemas de segurança que atenuam potenciais vulnerabilidades nesta solução, uma vez que sua atuação na arquitetura SDN expõe o arcabouço às ameaças emergentes desse paradigma. Portanto, neste capítulo serão abordadas as principais vulnerabilidades, riscos e ameaças relacionadas ao CAARF-SDN e ao ambiente de rede para o qual foi projetado.

4.2 VULNERABILIDADES E AMEAÇAS NA ARQUITETURA CAARF-SDN

Como já abordado no capítulo 3 desta dissertação, o arcabouço CAARF-SDN é proposto como uma solução dinâmica de encaminhamento de tráfego em redes SDN, baseado na programação automatizada do dispositivo Controlador. A aplicação CAARF-SDN atua no monitoramento e na coleta de informações sobre o contexto da rede, efetuando a otimização do tráfego por meio da definição de parâmetros de configuração enviados ao Controlador SDN, com o objetivo de reprogramar os fluxos dos comutadores *OpenFlow* da rede (BADARÓ NETO et al., 2018). Devido a esse processo de interação, as vulnerabilidades existentes no contexto das redes SDN interferem direta ou indiretamente no funcionamento do CAARF-SDN, e podem comprometer os recursos e serviços do arcabouço e conseqüentemente de toda a rede, uma vez que os ativos críticos da rede se concentram no Controlador SDN e no arcabouço CAARF-SDN.

Buscando avaliar o impacto de tais vulnerabilidade em relação ao arcabouço CAARF-SDN, são contemplados alguns cenários baseados em ameaças identificadas e catalogadas na Tabela 1.

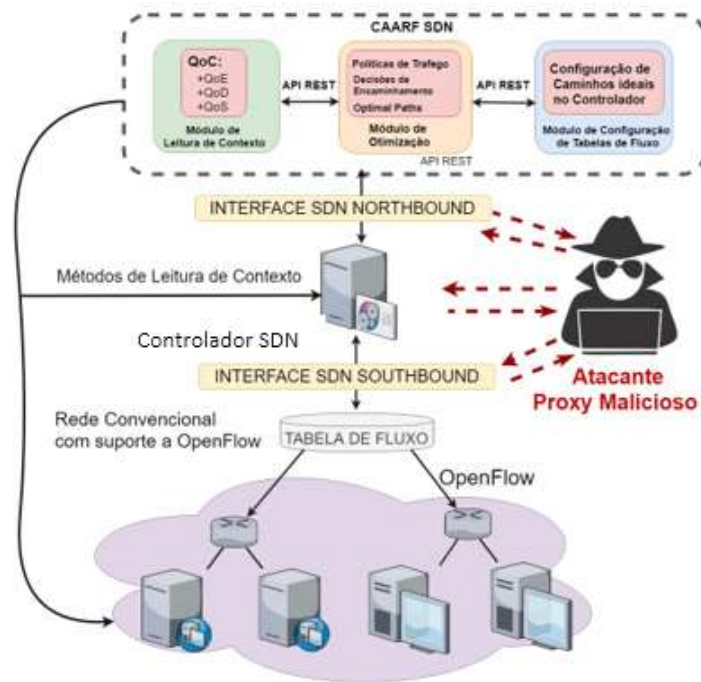
Tabela 1 – Vulnerabilidade e Ameaças em SDN

CAMADA	VULNERABILIDADE	ATAQUE
Controle/Aplicação	Ausência de mecanismos para lidar com conflitos de regras de fluxo	Falsificação de regras de fluxo.
		Manipulação de políticas de segurança.
	Aplicativos de rede não confiáveis	<i>Negação de Serviço (DoS)</i>
Esgotamento de recursos do sistema operacional de rede		
Controle/ Dados	Versão não segura de TLS nos dispositivos SDN	<i>Man in The Middle (Homem no meio)</i>
		Falsificação de mensagens OpenFlow
		Modificação das regras de fluxo
Controle	Ausência ou mecanismos de autenticação fracos	Invasão por aplicativos maliciosos (<i>malwares</i>)
	Ausência de controle de largura de banda do canal de comunicação <i>OpenFlow</i>	DoS / DDoS (ICMP Flood, SYN Flood etc).
	Ausência de mecanismos de identificação de fluxo	<i>Man In The Middle</i>
Dados	Ausência ou mecanismos de autenticação fracos	<i>Man In The Middle</i>
	Ausência de mecanismo de controle de fluxo adequado	DoS / DDoS (ICMP Flood, SYN Floode etc); <i>Man In The Middle.</i>

4.2.1 Ataques *Man In The Middle* no CAARF-SDN

Os sistemas operacionais de rede (*Network Operation System* - NOS) responsáveis por controlar o comportamento da rede *SDN*, por padrão, não são configurados para prover segurança das informações transmitidas e recebidas, possibilitando a interceptação do tráfego das interfaces (*Northbound* e *Southbound*) por meio de métodos de escuta na rede como ferramentas de *sniffers* (ABDELSALAM; EL-SISI, 2015). Para Balagopal e Rani (2018), as ações de um usuário malicioso podem ir além da captura do tráfego nas *interfaces Northbound* e *Southbound*, sendo possível a adulteração dos dados, sejam eles do sistema ou do contexto.

O sucesso desse tipo de ataque em uma rede *SDN* pode ocasionar riscos críticos ao funcionamento da rede e seus usuários, considerando que o invasor tem a possibilidade de monitorar, ler e alterar dados importantes. Devido à relação existente entre o CAARF-SDN e a arquitetura dos ambientes de redes definidas por software, esse tipo de ataque pode afetar o CAARF-SDN de diferentes maneiras: (i) as mensagens de notificação de contexto coletadas pelo módulo leitor de contexto do CAARF-SND podem ser modificadas através da interceptação e alteração de informações nos pacotes; (II) as diretivas de configuração enviadas pelo módulo de configuração "*Flowtable*" também podem ser interceptadas e modificadas antes de configurar a tabela de fluxo do controlador; (III) as regras de fluxo configuradas dentro da tabela de fluxo do controlador *SDN* podem ser modificadas e; (IV) as mensagens do *OpenFlow* podem ser forjadas quando acessadas pelos *switches* (HAYWARD; O'CALLAGHAN; SEZER, 2013). A Figura 14 ilustra o esquema de um ataque *Man In The Middle* no ambiente de rede híbrida do CAARF-SDN.

Figura 14 – Ataque *Man In The Middle* na rede CAARF-SDN

Fonte: Elaboração própria do autor desta dissertação (2019).

No ataque ilustrado, um usuário mal intencionado pode escutar o tráfego nas interfaces *Northbound* e *Southbound* de forma passiva, apenas analisando o conteúdo dos pacotes da rede através de ferramentas como *wireshark* ou de forma ativa, impersonificando dispositivos válidos como o Controlador SDN ou o CARRF-SDN, através de técnicas de clonagem MAC (*Media Access Control*) *address* ou comprometimento de serviços fundamentais da rede (*Domain Name System* - DNS, *Address Resolution Protocol* – ARP e Proxy), possibilitando a manipulação e falsificação das mensagens e instruções de controle do tráfego da rede (YOON et al., 2017).

A possibilidade de um invasor inserir, adulterar ou excluir as diretivas de configuração enviadas do arcabouço CAARF-SDN para o Controlador SDN através da *interface Northbound* é um risco crítico para o funcionamento da rede como um todo, afetando diretamente o comportamento correto dos dispositivos SDN (YOON et al., 2017).

Da mesma forma, um ataque de *Man In The Middle* na *interface Southbound* pode afetar a integridade do conteúdo da tabela de fluxo (*flowtable*), levando à configuração incorreta dos dispositivos SDN (YOON et al., 2017).

A ausência de soluções baseadas em criptografia e controle de tráfego estão diretamente relacionadas a esse tipo de vulnerabilidade em ambientes de redes SDN (PORRAS et al., 2015). Algumas estratégias e medidas são propostas na literatura a fim elevar o nível da segurança contra esse tipo de ataque (CORRÊA et al., 2016; SHU et al., 2016). A implementação de soluções de criptografia é amplamente discutida uma vez que a falta de mecanismos para implementação de camadas TLS/SSH nos canais de comunicação *Northband* e *Southband*, possibilitam a leitura e alteração das informações interceptadas de forma simples (SHU et al., 2016).

Mecanismos de controle de tráfego também são abordados na literatura como possíveis soluções no provimento de monitoramento, inspeção e controle de permissões dentro de ambientes SDN. Para esta finalidade, são apresentadas constantemente na literatura, a utilização de ferramentas tais como FortNox (PORRAS et al., 2012), VeriFlow (KHURSHID et al., 2012) e Flowchecker (SHU et al., 2016). Portanto, o controle de tráfego e a implementação de camadas TLS/SSH são abordadas como potenciais soluções para ataques do tipo *Man In The Middle* dentro de ambientes de rede SDN padrões ou baseados nessa arquitetura, como o CAARF-SDN. No entanto, é necessário destacar que mecanismos baseados em criptografia devem ser analisados cuidadosamente, uma vez que podem afetar o desempenho, que no ambiente de redes CAARF-SDN é um requisito fundamental para a interação em tempo real entre o arcabouço e o controlador SDN.

4.2.2 Negação de Serviço (*Denial of Service - DoS*) no CAARF-SDN

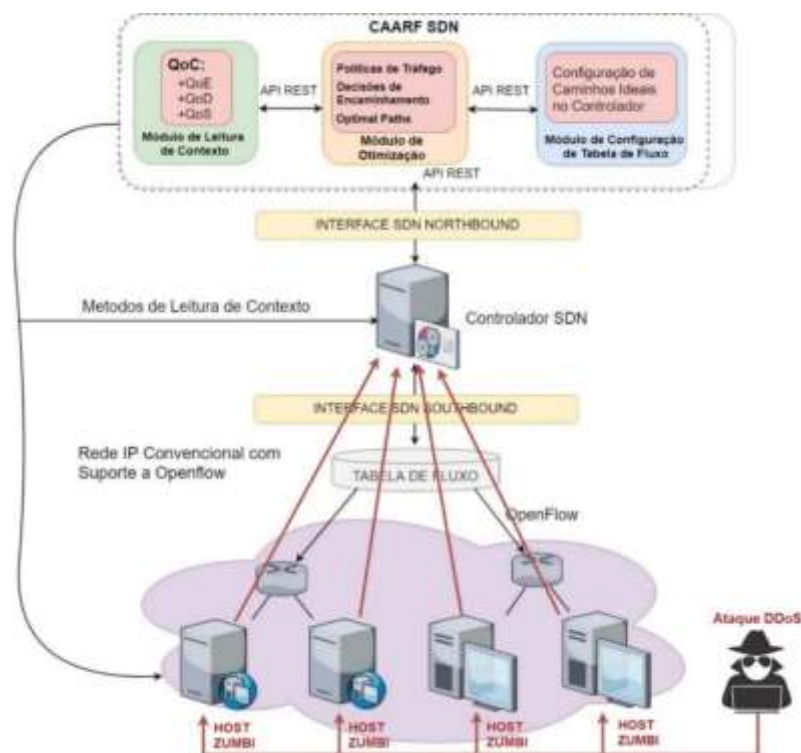
Esse cenário está relacionado a ataques que afetam a disponibilidade dos recursos da rede. Os ataques desse nível podem ser classificados em dois tipos: Negação de Serviço (*Denial of Service - DoS*) e Negação de Serviço Distribuído (*Distributed Denial of Service - DDoS*) (CLARO, 2015). Ataques do tipo DoS são realizados de forma não distribuída, em uma escala menor (local), a partir de um único dispositivo, já os ataques de *Distributed Denial of Service* (DDoS) funcionam de forma distribuída onde o atacante se utiliza de diversos dispositivos, para sobrecarregar os serviços da rede com milhares de falsas requisições de conexão. Em ambos os casos, o atacante sequestra dispositivos válidos da rede ou remotos e utiliza esses hospedeiros para executar ataques em conjunto contra os alvos (CERT.BR, 2012). Nas redes SDN esse tipo de ataque representa um risco crítico para o Controlador da rede, que pode ser um alvo fácil de um ataque bem planejado (BENTON; CAMP; SMALL, 2013). No caso específico do CAARF-SDN, a disponibilidade dos recursos e serviços é uma questão

ainda mais crítica, uma vez que existem dois pontos vulneráveis de extrema importância para o funcionamento da rede, o Controlador SDN e o arcabouço CAARF-SDN. Nesse contexto, um ataque de negação de serviço pode afetar de forma significativa o arcabouço CAARF-SDN ou ainda o Controlador, interrompendo serviços fundamentais da rede (SHU et al., 2016).

Nesta sessão é apresentado na Figura 15 um cenário conceitual para ilustrar como um ataque desse tipo pode afetar vários hospedeiros da camada de dados, utilizando esses dispositivos para realizar uma série de falsas requisições para o controlador SDN, através de inundação de pacotes, sobrecarregando sua capacidade de atender as necessidades da rede, conseqüentemente, levando a sua indisponibilidade. É importante notar que o mesmo ataque pode ser realizado contra o arcabouço CAARF-SDN, com o objetivo de interromper o seu funcionamento.

Ataques dessa natureza podem ocorrer localmente ou remotamente tirando proveito de vulnerabilidades já citadas anteriormente, tais como a falta de mecanismos para controle de tráfego nos canais de comunicação (*Northbound* e *Southbound*) ou mecanismos fracos de autenticação, assim como configuração inadequada em dispositivos da rede (SHU et al., 2016; YOON et al., 2017).

Figura 15 – Ataque DDoS na rede CAARF-SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

Ataques DoS ou DDoS são difíceis de detectar em ambientes sem recursos para identificação e tratamento desse tipo de vulnerabilidade. A fim de evitar esses ataques dessa natureza, duas soluções principais são propostas na literatura: a implementação do controle de tráfego e redundância de dispositivos da rede (CORRÊA et al., 2016).

Com a implementação do controle de tráfego é possível identificar e filtrar o tráfego de acordo com sua origem, destino, serviço, etc. Neste caso, as boas práticas introduzidas na literatura são baseadas ainda na limitação da taxa de transmissão dentro dos canais de comunicação *OpenFlow* entre camadas de aplicação, controle e dados visando prevenir o tráfego de inundação de pacotes direcionados ao Controlador SDN evitando sua indisponibilidade (CORRÊA et al., 2016). As ferramentas apresentadas na literatura aplicadas em geral para a implementação do controle de tráfego em redes SDN são FortNox (PORRAS et al., 2012), VeriFlow (KHURSHID et al., 2012), FlowChecker (PORRAS et al., 2015), entre outras.

Outra estratégia de segurança aplicada com o objetivo de evitar ataques DDoS é a implementação de redundância dos principais serviços na plataforma SDN, como o controlador. No ambiente do CAARF-SDN, o arcabouço também é considerado nessa abordagem de segurança. Por exemplo, uma possível solução é apresentada em (SHU et al., 2016) é a implementação de três controladores SDN distribuídos fisicamente, mas logicamente centralizados, trocando informações de forma colaborativa sobre a topologia e o estado geral da rede, através de duas novas interfaces (*Eastbound* e *Westbound*). Nesta abordagem, é importante considerar a compatibilidade do controlador SDN com recursos de espelhamento colaborativo. Atualmente, os controladores de SDN tais como o ONOS (PORRAS et al., 2015), OpenDayLight (ROTHENBERG et al., 2012) e o NOX (YOON et al., 2017) são compatíveis com estas características. É interessante notar que esta abordagem pode ser aplicada de forma semelhante à camada de aplicação pela implementação de redundância dos serviços do CAARF-SDN. Nesse caso, a descentralização física desses serviços podem proporcionar escalabilidade, disponibilidade e a efetividade dos serviços da CAARF-SDN (SHU et al., 2016).

4.2.3 Softwares Maliciosos (*Malwares*) no CAARF-SDN

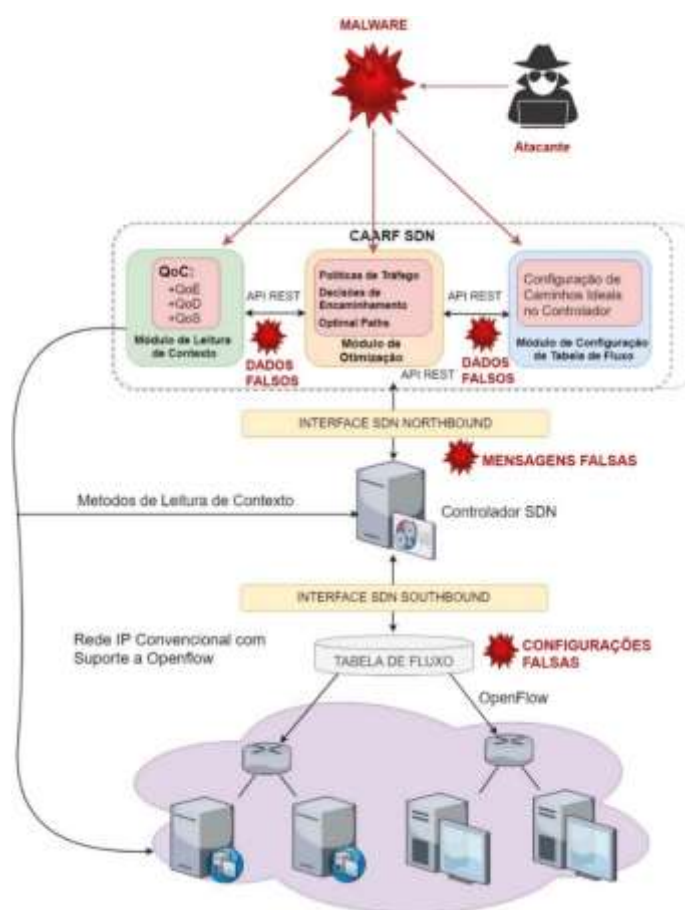
Este cenário ilustra o ataque realizado por softwares maliciosos (*malware*). Os *Softwares* malicioso podem atacar a integridade de informações em ambientes de redes *SDN* através do comprometimento da camada de aplicação, infectando o controlador (SHU et al., 2016; PORRAS et al., 2015; ONF, 2013). No ambiente híbrido do CAARF-SDN as informações de

contexto também podem ser alvos de falsificação ou adulteração, levando à definição incorreta do contexto e, conseqüentemente, à geração de configuração de otimização incoerentes.

Esse tipo de ataque afeta diretamente à confidencialidade e à integridade dentro da rede, já que todas as informações de contexto podem ser interceptadas e adulteradas pelo invasor (MIGAULT; POURZANDI, 2015; RÖPKE; HOLZ, 2015). A falta de mecanismos de autenticação e criptografia permitem que invasores explorem essas vulnerabilidades (PORRAS et al., 2015; RÖPKE; HOLZ, 2015; SHU et al., 2016).

A Figura 16 ilustra de forma conceitual esse tipo de ataque no contexto de rede do arcabouço CAARF-SDN.

Figura 16 – Ataque *malwares* na rede CAARF-SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

Os *malwares* podem ainda ser utilizados para dar acesso privilegiado ao controlador, proporcionando ao usuário malicioso o controle total da rede, assim como, possibilitando a reprogramação das tabelas de fluxo dos *switches*, inserindo fluxos falsos. Ferramentas como SEFloodlight (PORRAS et al., 2015), FRESCO (LOPEZ, 2014) e PermOF (HAYWARD;

NATARAJAN; SEZER, 2016) são propostas na literatura como mecanismos de contenção de ataques *malwares* no contexto das redes definidas por software. As técnicas de defesa utilizam uma abordagem baseada em autenticação de aplicativos que podem ser identificados como confiáveis ou não, através da avaliação de seu controle de acesso, a fim de permitir que eles modifiquem ou não a tabela de fluxo SDN dentro de *switches OpenFlow*, limitando a utilização dos recursos do controlador (PORRAS et al., 2015; HAYWARD, 2015b). Essa abordagem impede que os *malwares* modifiquem as políticas válidas estabelecidas para o ambiente de rede SDN, impondo níveis de restrição na camada de aplicação, e assegurando aspectos de segurança como integridade, autenticidade, confidencialidade e disponibilidade de rede (LOPEZ, 2014). Esses recursos afetam diretamente o CAARF-SDN, uma vez que o mesmo atua na camada de aplicativo da rede SDN, tornando-se vulnerável a *softwares* maliciosos.

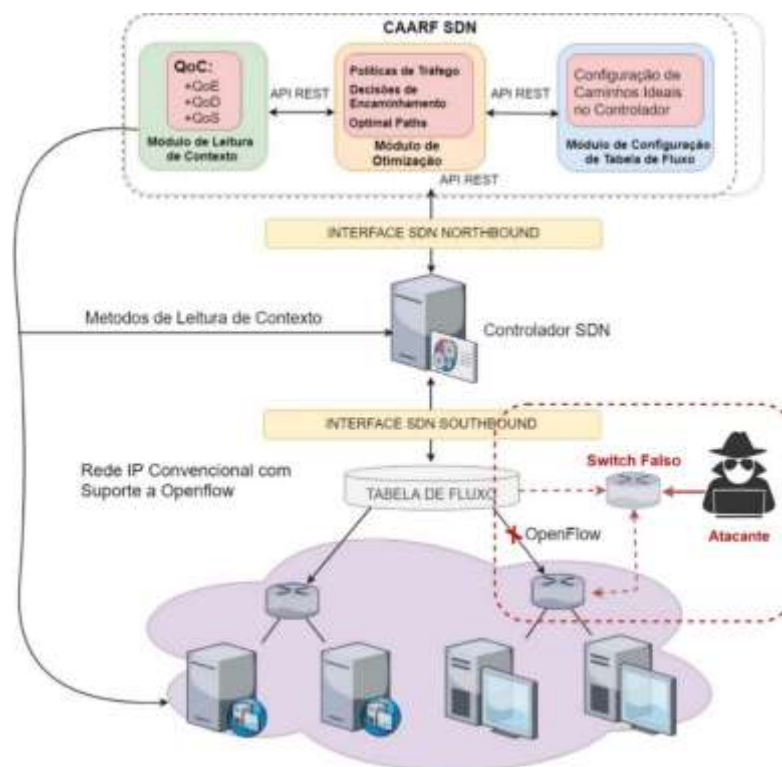
4.2.4 Ataques de Falsificação de *Switch OpenFlow* no CAARF-SDN

A possibilidade de inserir dispositivos de rede forjados é discutida neste cenário. Em particular, a Figura 17 ilustra a possibilidade de um invasor introduzir um comutador *OpenFlow* falso em uma rede baseada em SDN, visando manipular tabelas de fluxos e políticas estabelecidas na rede. Neste caso, o dispositivo forjado personifica um dispositivo autêntico depois de tê-lo bloqueado (por exemplo, através de um ataque DDoS) (CORRÊA et al., 2016; ROTHENBERG et al., 2012).

O ataque ocorre através da clonagem do endereço físico do *switch OpenFlow* válido, onde o dispositivo forjado passa a receber todas as atualizações de fluxo destinadas ao *switch* original. Além da clonagem, é possível ainda atribuir um endereço físico (MAC) aleatório para o dispositivo válido, fazendo com que esse dispositivo deixe de ser enxergado como um dispositivo da rede pelo Controlador (YOON et al., 2017).

Como consequência desse tipo de ataque, as diretivas de comutação configuradas no controlador SDN podem ser ignoradas pelo dispositivo forjado que conduz à rede a um funcionamento incorreto. O dispositivo forjado ainda pode ser utilizado para propagar diretivas de comutação falsa para os demais *switches OpenFlow* válidos e até mesmo interceptar informações relevantes da rede e de toda a topologia (SHU et al., 2016).

Figura 17 – Ataque de Falsificação de *Switches OpenFlow* na rede CAARF-SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

Os ataques deste tipo são possíveis devido à falta de mecanismos de autenticação, ou devido à configuração de mecanismos fracos ou incorretos. Além disso, a falta de mecanismos de controle de tráfego pode igualmente permitir que estas vulnerabilidades sejam exploradas (PORRAS et al., 2015; RÖPKE; HOLZ, 2015).

Nesse cenário, as principais estratégias de segurança propostas e discutidas na literatura são mecanismos de autenticação estão relacionadas à utilização de softwares como VeriFlow, FlowVisor e FlowChecker, assim como à implementação de sistemas de IDS/IPS, como SNORT e Suricata (CAMELO et al., 2017). Essas abordagens impedem a rede de intrusão e das tentativas de forjar dispositivos de aplicação, controle e camada de dados.

4.3 CONCLUSÃO

Nesse capítulo foram propostos modelos conceituais propostos com o objetivo de analisar as principais vulnerabilidades identificadas no contexto das redes SDN, assim como o impacto das ameaças no comportamento do CAARF-SDN. Foram discutidos ainda, algumas

possíveis soluções propostas na literatura com o objetivo de mitigar vulnerabilidades em ambientes de redes baseados na arquitetura SDN.

As análises levantadas viabilizam os recursos necessários para a realização dos experimentos que foram propostos nessa dissertação, como contribuição para o projeto CAARF-SDN, que são apresentados no próximo capítulo.

5 SOLUÇÕES PROPOSTAS

5.1 INTRODUÇÃO

Apesar do CAARF-SDN ser uma estrutura em parte independente, a viabilidade e sucesso de sua implantação e funcionamento em uma rede SDN está altamente relacionada com todos os elementos existente no contexto, principalmente o Controlador SDN. Sendo assim, é importante destacar que as soluções propostas no presente trabalho, não se destinam exclusivamente ao arcabouço do CAARF-SDN, mas sim a todo o contexto do ambiente de rede focado no paradigma SDN em que o arcabouço está inserido, uma vez que seria impossível reduzir os riscos identificados, focando apenas no arcabouço e ignorando todos os demais ativos influentes na rede SDN.

Neste capítulo serão apresentados e descritos os experimentos utilizados nessa dissertação, com o objetivo de analisar, implementar e validar estratégias e mecanismos de segurança no ambiente de rede do arcabouço CAARF-SDN.

O foco dos experimentos realizados, se norteiam pela análise do comportamento das vulnerabilidades e ataques identificados no capítulo 4 desse trabalho, que representam risco ao funcionamento adequado do cenário de rede SDN, assim como a interação dos seus componentes com o arcabouço CAARF-SDN. Posteriormente na implementação e validação de estratégias e dos mecanismos de análise de tráfego e prevenção de ataques, buscando garantir aspectos de segurança como integridade e disponibilidade no ambiente de redes do arcabouço e seus serviços.

5.2 CARACTERIZAÇÃO DO EXPERIMENTO

As estratégias e mecanismos implementados nos experimentos desse estudo se baseiam em dois cenários virtualizados, que tem por finalidade, explorar as vulnerabilidades e falhas que podem colocar em risco o funcionamento adequado da rede e do próprio arcabouço CAARF-SDN. Basicamente os cenários representam ambientes de redes SDN operacionais e propício a integração com o arcabouço CAARF-SDN.

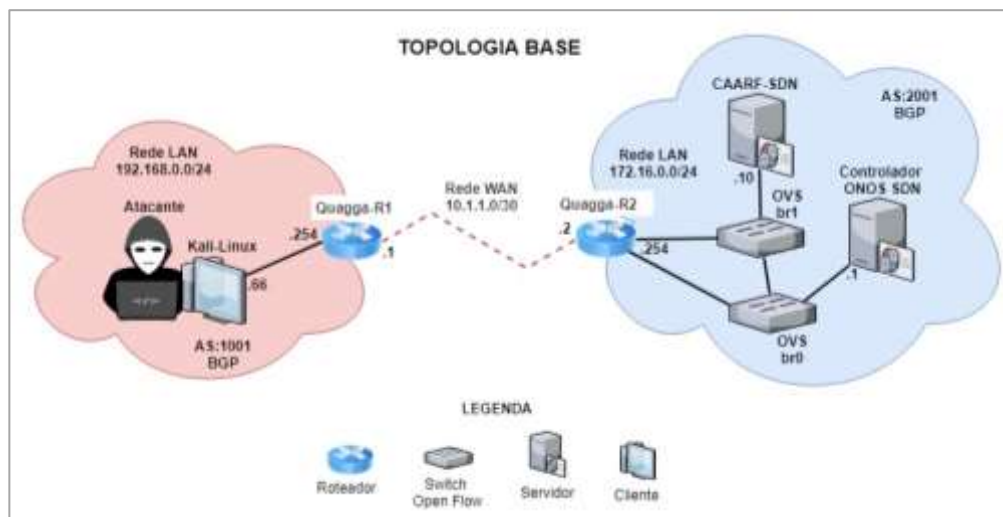
O principal objetivo dos experimentos em primeiro momento, foi avaliar o comportamento do ambiente de rede proposto para o arcabouço CAARF-SDN em relação a ataques e vulnerabilidades relacionadas diretamente com aspectos de segurança, principalmente a disponibilidade dos ativos críticos da rede. Posteriormente é feita uma análise dos resultados obtidos após a aplicação das soluções para mitigação de ataques focados na

disponibilidade, buscando medir o seu impacto dessas medidas no escopo global da rede, principalmente nos dispositivos críticos da rede, a saber: arcabouço CAARF-SDN e Controlador SDN.

5.2.1 Topologia Utilizada nos Cenários

A topologia utilizada nos experimentos dessa dissertação, será a mesma para ambos os cenários apresentados, distinguem-se apenas na aplicação das soluções e mecanismos de segurança aplicados. A Figura 18 apresenta a topologia base da rede, desenvolvida especificamente para esse trabalho.

Figura 18 – Topologia base dos cenários propostos



Fonte: Elaboração própria do autor desta dissertação (2019).

Basicamente a topologia é formada por duas redes distintas interligadas por roteadores de borda executando a comunicação através do protocolo BGP (*Border Gateway Protocol*), que por sua vez viabiliza a troca de informações entre Sistemas Autônomos (*Autonomous System - AS*), que nada mais são do que “redes autônomas interconectadas” (VALCY; RIBEIRO; SAMPAIO, 2018) seguindo as mesmas políticas de gerência técnica e compartilhando políticas internas e externas bem definidas de roteamento. Essa estrutura foi escolhida para simular adequadamente, situações de acesso indevido do dispositivo atacante aos dispositivos do ambiente de rede do CAARF-SDN, sendo que os experimentos simulam ataques originados em uma rede externa ao arcabouço, com o objetivo de analisar o ataque realizado de forma remota.

É importante destacar que protocolo BGP foi adotado nessa topologia com o objetivo de viabilizar a interconexão entre as redes apresentadas na topologia, uma vez que esse protocolo tem suporte nativo por parte dos softwares controladores de redes SDN, que é amplamente utilizado com sucesso na interconexão de Redes Definidas por Softwares (VALCY; RIBEIRO; SAMPAIO, 2018). Essa abordagem tem a finalidade de aproximar os experimentos de situações do mundo real e facilitar a documentação da topologia adotada. A Tabela 2 apresenta os detalhes da topologia adotada nesse experimento.

Tabela 2 - Descrição da topologia base dos experimentos de rede

Dispositivo	IP	Mascara	AS BGP	Tipo	Links
Atacante 01	LAN=192.168.0.66	/24	1001	Comp. Cliente	Quagga-R1
Atacante 02	LAN=192.168.0.67	/24	1001	Comp. Cliente	Quagga-R1
Quagga-R1	LAN=192.168.0.254	/24	1001	Roteador	Quagga-R2
	WAN=10.1.1.1	/30			
Quagga-R2	LAN=172.16.0.254	/24	2001	Roteador	Quagga-R1
	WAN=10.1.1.2	/30			
OVS-BR0	-----	-----	2001	Switch OpenFlow	Controlador Quagga-R2
OSV-BR1	-----	-----	2001	Switch OpenFlow	CAARF- SDN Quagga-R2
Controlador ONOS SDN	LAN=172.16.0.1	/24	2001	Comp. Servidor	OVS-BR0
CAARF- SDN	LAN=172.16.0.10	/24	2001	Comp. Servidor	OVS-BR1

As informações apresentadas na Tabela 2, destacam a especialização de cada dispositivo dentro do seu respectivo AS, assim como seus endereços e *links*.

A rede do atacante, representada pelo AS 1001, é definida de forma geral através do endereçamento de rede 192.168.0.0/24 e o dispositivo malicioso é identificado especificamente pelo IP 192.168.0.66/24. A rede do atacante está conectada a um roteador de borda que funciona como *gateway* de saída para a rede WAN (*Wide Area Network*) identificada pelo endereço 10.1.1.1/30. Na outra ponta da topologia, temos o ambiente de rede no qual o arcabouço CAARF-SDN está inserido. Essa segunda rede da topologia é identificada de forma geral pelo endereçamento 172.16.0.0/24 e está representada pelo AS 2001, que será utilizado

para estabelecer vizinhança com o AS 1001. Existem ainda dois *switches OpenFlow* que fazem a comutação entre o arcabouço CAARF-SDN e o Controlador SDN, assim como o encaminhamento dos fluxos SDN aplicados pelo controlador.

Outra característica importante para o funcionamento adequado dos experimentos dessa dissertação está nos requisitos de *hardware*, tendo em vista as necessidades dos dispositivos em relação a recursos como processamento, armazenamento, memória de leitura e entre outros. A Tabela 3 apresenta os detalhes de *hardware* de todos os dispositivos da topologia apresentada na Figura 15.

Tabela 3 - Configuração de hardware dos dispositivos da topologia

Dispositivo	S.O	Núcleos da CPU	Memória	HD	Placa de Rede
Atacante	Kali-Linux	1	4 GB	20 GB	1
Quagga-R1	Debian 9	1	512 MB	10 GB	2
Quagga-R2					
OVS-BR0	Ubuntu 16	1	2 GB	10 GB	6
OVS-BR1					
Controlador SDN	Unbutu 16	2	4 GB	20 GB	1
CAARF-SDN	Ubuntu 16	1	2 GB	20 GB	1
Host Zumbi 1	Debian 8	1	256 MB	10 GB	1
Host Zumbi 2					
Host Zumbi 3					

Os requisitos de hardware apresentados, assim como os sistemas operacionais utilizados, são resultantes de diversos testes de desempenho realizados ao longo desse trabalho, com o objetivo de promover um ambiente capaz de atender as necessidades de hardware necessárias para que cada dispositivo, execute os softwares exigidos e desempenhem seu papel adequadamente dentro da topologia.

5.2.2 Definição das Aplicações Utilizadas

Além da virtualização, outras ferramentas foram implementadas para proporcionar a execução de todos os elementos do cenário de rede proposto na topologia apresentada, a saber:

- a) **VMware Workstation:** A ferramenta escolhida para virtualizar os componentes de redes utilizados na topologia apresentada na sessão 5.1.1 foi o VMware Workstation 14, devido a quantidade de ferramentas disponibilizadas pelo software, assim sua arquitetura de alto desempenho em comparação com outros softwares da mesma categoria. Possui uma interface amigável e de fácil interação, além de disponibilizar uma diversidade de recursos que facilitam a organização e o gerenciamento das VMs e de todos os seus componentes (CPU, memória RAM, HD, rede e etc.) (RODRIGUES; SILVA; LEITE, 2011);
- b) **GNS 3:** O Simulador Gráfico de Redes, ou GNS3 (*Graphical Network Simulator 3*) como é conhecido, trata-se de um software que permite a criação de topologias de rede complexas através de *interface* gráfica (GNS3, 2019). O GNS3 foi utilizado nos experimentos desse trabalho, com o objetivo de criar uma topologia gráfica do ambiente virtualizado, uma vez que o *software* tem suporte nativos para interação com o VMware Workstation;
- c) **ONOS (*Open Networking Operating System*):** É um controlador SDN de código aberto que fornece um plano de controle para Redes Definidas por *Software*, viabilizando o gerenciamento dos dispositivos da rede (comutadores, *links*, *hosts* e entre outros) e “executando programas ou módulos de software para fornecer serviços de comunicação para hosts finais e redes vizinhas” (ONOS, 2019). Uma das grandes vantagens do ONOS é a possibilidade de ser implementado e executado de forma distribuída, criando um *cluster* na camada de controle para fornecer recursos de tolerância a falhas (ONOS, 2019). O núcleo do ONOS, assim como seus principais serviços e aplicativos são desenvolvidos sob a tecnologia JAVA, dessa forma fornece interoperabilidade, possibilitando a execução e interação do ONOS com diversas plataformas (ONOS, 2019).
- d) **Quagga:** Esse é um *software Open Source* para roteamento em redes computacionais, amplamente utilizado no provisionamento de diversos

protocolos de rede na plataforma Unix (*Linux, BSD, Solaris* e etc.) como RIP, OSPF e BGP (JAKMA; LAMPARTER, 2014). O Quagga foi utilizado para criar e estabelecer vizinhança entre os sistemas autônomos (AS) através do protocolo BGP das redes distintas propostas nos experimentos dessa dissertação, possibilitando a comunicação entre a rede do atacante e do CAARF-SDN, uma vez que o protocolo BGP é o padrão de comunicação inter-AS adotado nativamente pelos softwares controladores das redes SDN;

- e) **Suricata IDS:** O Suricata é uma solução de IDS baseada em *software* livre, que usa um conjunto de regras (padrões ou personalizadas) bem definidas para monitorar o tráfego, capturando pacotes e buscando por assinaturas de possíveis invasões da rede, sendo essas assinaturas definidas como padrões ou comportamentos que identificam um tráfego suspeito (SURICATA, 2019).
- f) O Suricata suporta uma diversidade de protocolos, sendo os mais comuns, reconhecidos automaticamente quando a captura do tráfego é iniciada. Além dos protocolos é possível identificar uma infinidade de extensões de arquivos durante as varreduras da rede, mapeando os metadados e identificadores dos mesmos (BERALDO et al., 2018; SURICATA, 2019). Todas essas características fazem do Suricata, uma ferramenta de segurança de redes robusta e indispensável para garantir aspectos fundamentais para mitigar ameaças e riscos, aumentando assim a confiabilidade, integridade e disponibilidade de ativos de rede.
- g) A implementação do Suricata dentro do presente projeto, tem por finalidade proporcionar um mecanismo de segurança com o objetivo de detectar de forma dinâmica, ataques direcionados ao ambiente de rede do arcabouço CAARF-SDN, tendo como prioridade a proteção do Controlador SDN e do arcabouço do CAARF-SDN. O processo de implementação e validação dessa ferramenta será abordado em sessões posteriores;
- h) **Guardian:** A ferramenta Guardian é uma solução de segurança que trabalha em conjunto com sistemas IDS como o Snort e o Suricata, com o objetivo de promover contenção automatizada de ameaças. Basicamente o Guardian atualiza regras no *firewall* com base nos com base nos alertas gerados pelos

IDS, bloqueando todo o tráfego recebido do endereço IP malicioso (CHAOTIC, 2002; VALCY; RIBEIRO; SAMPAIO, 2018). O Guardian foi utilizado em conjunto com o Suricata IDS como solução para provisionar reação automatizada a ameaças no ambiente de rede do arcabouço CAARF-SDN;

- i) **Elastic Stack:** Pode ser definido como um conjunto de ferramentas que implementam o conceito de *Big Data* para tratar grandes quantidades de dados estruturados e não estruturados em tempo real, obtidos de qualquer fonte (CHHAJED, 2015). Através das ferramentas do Elastic Stack é possível realizar desde a centralização de *logs* até o rastrear métricas de infraestrutura (CHHAJED, 2015). As ferramentas serão implementadas no experimento dessa dissertação para viabilizar uma *interface* gráfica de monitoramento e alerta baseado na leitura dos logs do Suricata IDS;
- j) **Atomix:** É definida como um *framework* que atua como solução na criação e gerenciamento de sistemas distribuídos. Através do Atomix é possível implementar e manter *clusters* consistentes em diversos tipos de sistemas (SPALLA et al., 2015; ATOMIX, 2019). O Atomix foi utilizado na implementação do cenário de alta disponibilidade do arcabouço CAARF-SDN proposto como parte dos experimentos realizados nessa dissertação, com o objetivo de prover alta disponibilidade dos dispositivos da rede CAARF-SDN, assim como seus recursos e serviços;
- k) **Wireshark:** É conhecido como um dos softwares mais utilizados na análise de tráfego de redes computacionais. O Wireshark possibilita a inspeção profunda dos pacotes (todas as informações) que trafegam entre os *hosts* dentro da rede (WIRESHARK, 2019);
- l) **Hping3:** De acordo com (HPING, 2016), o Hping3 é um *software* de licença livre, focado na criação de pacotes IP, com o propósito de testar *firewalls* e sistemas de detecção de intrusão em redes computacionais. Através do Hping3 é possível realizar varreduras de sistemas (*portscam*), gerar tráfego intenso de dados e de modo geral, criar pacotes arbitrariamente conforme as necessidades de análise da rede (GOULART et al., 2015). A ferramenta funciona no modo linha de comandos (*Command-Line Interface* - CLI) e trabalha com diversos tipos de protocolo como por exemplo TCP

(*Transmission Control Protocol*), UDP (*User Datagram Protocol*) e ICMP (*Internet Control Message Protocol*). Outra grande característica do Hping3 é a possibilidade de modificar os cabeçalhos dos pacotes gerados (BOGAERTS; XXRADAR, 2003).

- m) **Common Vulnerability Scoring System (CVSS)**: É definido como uma estrutura aberta para medir e avaliar o nível de gravidade de vulnerabilidades em sistemas. Como apresentado em (CVSS, 2019), o CVSS “consiste em três grupos de métricas: Base, Temporal e Ambiental”. O grupo “Base” está relacionado com vulnerabilidades que são “constantes ao longo do tempo nos ambientes do usuário”. O grupo “Temporal” faz referência as características de vulnerabilidades que podem variar de acordo com o tempo. Por fim, o grupo Ambiental reflete a forma como o ambiente onde se encontra a vulnerabilidade influencia na exploração da falha e na gravidade do risco (CVSS, 2019). O sistema CVSS será utilizado ao final dos experimentos desse trabalho, para medir o nível dos riscos e vulnerabilidades antes e depois da implementação dos mecanismos de segurança propostos para o ambiente de rede do CAARF-SDN, com objetivo de auxiliar na validação das soluções propostas nessa dissertação.

A Tabela 4 apresenta todas as aplicações utilizadas em todos os processos dos experimentos realizados nesse trabalho.

Tabela 4 – Aplicações utilizadas

Software	Versão	Sistema
Vmware Workstation	14.0	Windows 10
GNS3	3.1.1	Windows 10
ONOS	1.15.0 – 2.0.0	Ubuntu 16 (VM)
CAARF-SDN	-----	Ubuntu 16 (VM)
Atomix	3.1.0-beta1	Ubuntu 16 (VM)
Quagga	2.9.8	Debian 9 (VM)
Suricata IDS/IPS	3.0.1	Ubuntu 16 (VM)
Guardian	1.0.7	Ubuntu 16 (VM)

Software	Versão	Sistema
Elastic	5.5.3	Ubuntu 16 (VM)
Logstash	5.5.3	Ubuntu 16 (VM)
Kibana	5.5.3	Ubuntu 16 (VM)
Wireshark	3.0.3	Kali-Linux (VM)
Hping3	3.0	Kali-Linux (VM)
CVSS	3.0	-----

5.3 CARACTERIZAÇÃO DOS ATAQUES DoS (*DENIAL of SERVICE*) REALIZADOS

O principal objetivo dos experimentos realizados nessa sessão, é a análise do ambiente de rede do CAARF-SDN sobre a perspectiva da disponibilidade dos recursos e serviços, uma vez que foi identificado nesse trabalho como uma das principais vulnerabilidades no contexto das redes SDN e do arcabouço CAARF-SDN. Os ataques DoS (*Denial of Service*) foram realizados tendo como alvo o Controlador SDN e o hospedeiro de referência do arcabouço CAARF-SDN, devido à importância e criticidade desses dispositivos dentro da rede, como foi identificado na análise de riscos da tabela CVSS apresentada no capítulo 4.

Os experimentos foram realizados em duas etapas, na primeira os ataques foram efetuados em um ambiente de rede *default*, sem nenhum mecanismo de segurança implementado, visando esgotar os recursos do Controlador SDN e do hospedeiro de referência do arcabouço CAARF-SDN, tornando-os indisponíveis para toda a rede. O tráfego das interfaces relevantes foi monitorado durante os experimentos, possibilitando a avaliação das consequências e a análise do comportamento da rede CAARF-SDN ao ser exposta a ataques dessa natureza.

5.4 CENÁRIO 1 - ATAQUES DoS SEM MECANISMOS DE CONTENÇÃO

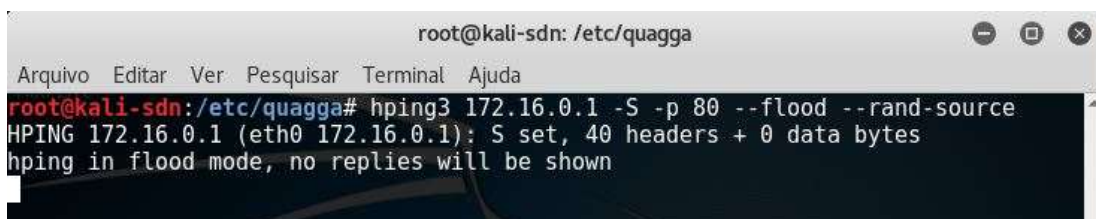
Esse cenário consiste na realização de ataques do tipo SYN Flood e ICMP Flood no ambiente de rede apresentado na sessão 5.2 como base para realização dos experimentos dessa dissertação. Os ataques foram realizados utilizando a ferramenta hping3 através do sistema operacional Kali-Linux.

O objetivo do experimento é sobrecarregar e exaurir os recursos do Controlador SDN e do hospedeiro de referência do CAARF -SDN com inúmeras requisições TCP SYN e ICMP, ocasionando falhas e indisponibilidade de serviços e recursos.

5.4.1 Ataques de SYN Flood e ICMP Flood na Rede CAARF-SDN

O primeiro ataque executando contra o hospedeiro do controlador ONOS e o hospedeiro de referência do arcabouço CAARF-SDN foi o SYN Flood. A Figura 19 apresenta os parâmetros do ataque SYN Flood sendo realizado através do dispositivo malicioso tendo como alvo o endereço IP do Controlador SDN.

Figura 19 – Ataque SYN Flood no Controlador SDN



```

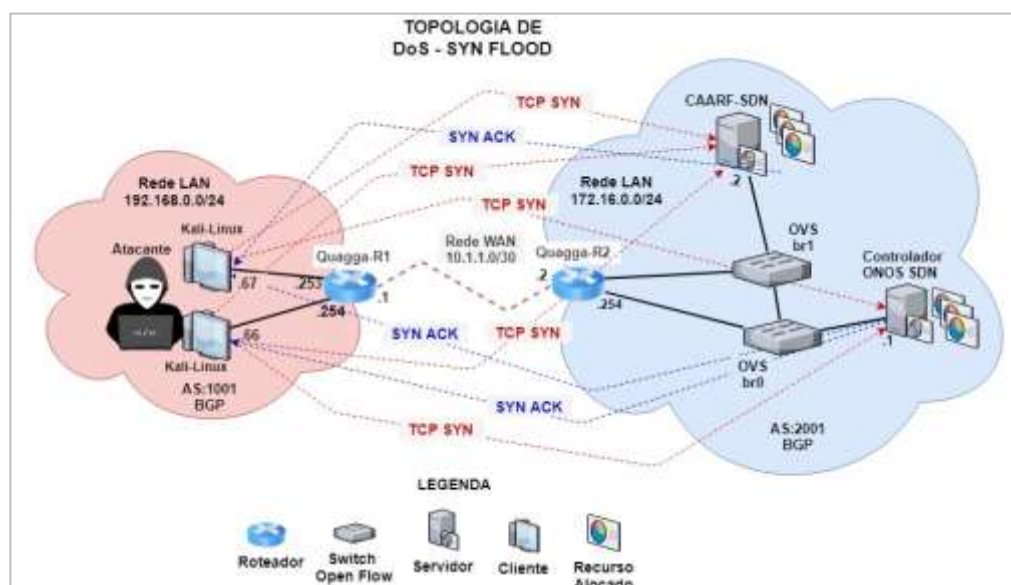
root@kali-sdn: /etc/quagga
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali-sdn:/etc/quagga# hping3 172.16.0.1 -S -p 80 --flood --rand-source
HPING 172.16.0.1 (eth0 172.16.0.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
  
```

Fonte: Elaboração própria do autor desta dissertação (2019).

Os parâmetros do ataque utilizados pela ferramenta hping3 possuem funções bem definida dentro da estrutura do comando. O “-S” indica que o ataque é do tipo SYN Flood, o “-p 80” promove o envio dos pacotes para a porta 80 e o “--*rand-source*” fornece endereços de IPs falsos e aleatório para camuflar o ataque através da técnica de IP Spoof.

A Figura 20 ilustra de forma conceitual do ataque SYN Flood no cenário de rede do CAARF-SDN.

Figura 20 – Modelo conceitual do ataque SYN Flood na rede CAARF-SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

O segundo ataque foi um tipo de ICMP Flood denominado de *Smurf*, com o objetivo de enganar os dispositivos válidos da rede com falsas requisições de *Echo Request*, redirecionando todas as respostas *Echo Response* para o hospedeiro CAARF-SDN e o Controlador ONOS. A Figura 21 apresenta os parâmetros em linha de comando utilizados para esse ataque.

Figura 21 – Ataque de ICMP Flood no Controlador SDN

```

root@kali-sdn: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali-sdn:~# hping3 --spooof 1.2.3.4 --icmp-ts 172.16.0.1
HPING 172.16.0.1 (eth2 172.16.0.1): icmp mode set, 28 headers + 0 data bytes

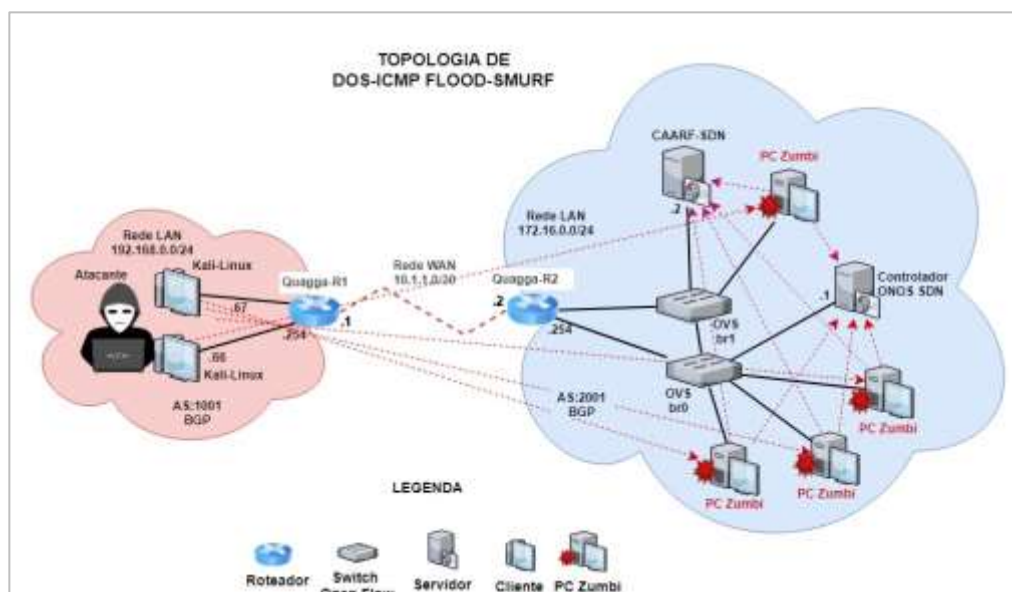
```

Fonte: Elaboração própria do autor desta dissertação (2019).

Basicamente o hospedeiro malicioso executa o ataque passando o endereço IP do Controlador SDN e do arcabouço CAARF-SDN como se fossem um endereço de broadcast enviando solicitações. Todos os demais hosts da rede recebem a requisição de *Encho Request* e respondem massivamente com um *Echo Response*, sobrecarregando dessa forma a interface de rede e esgotando recursos de processamento do hospedeiro alvo.

A Figura 22 ilustra o ataque *Smurf* na rede do arcabouço CAARF-SDN, possibilitando uma melhor compreensão da extensão do ataque.

Figura 22 – Modelo conceitual do ataque *Smurf* na rede CAARF-SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

Nesse tipo de ataque, os hospedeiros “zumbis” participam do ataque de forma involuntária, aumentando o potencial da ameaça ocasionar falhas graves na rede. Os detalhes do ataque podem ser observados através da ferramenta Wireshark que foi executada para monitorar de forma mais profunda todo o tráfego gerado para o arcabouço CAARF-SDN como pode ser visto na Figura 23.

Figura 23 – Captura do Ataque SYN Flood no Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
37933	87.367474931	51.63.185.200	172.16.0.1	TCP	54	5896
37934	87.371160767	182.192.8.13	172.16.0.1	TCP	54	5897
37935	87.372146016	200.221.142.51	172.16.0.1	TCP	54	5897
37936	87.372561395	12.17.146.226	172.16.0.1	TCP	54	5897
37937	87.373080885	28.69.88.145	172.16.0.1	TCP	54	5897
37938	87.377028089	185.17.241.76	172.16.0.1	TCP	54	5897
37939	87.377566767	29.18.171.166	172.16.0.1	TCP	54	5897
37940	87.378020178	30.183.29.166	172.16.0.1	TCP	54	5897
37941	87.378420827	251.03.148.249	172.16.0.1	TCP	54	5897

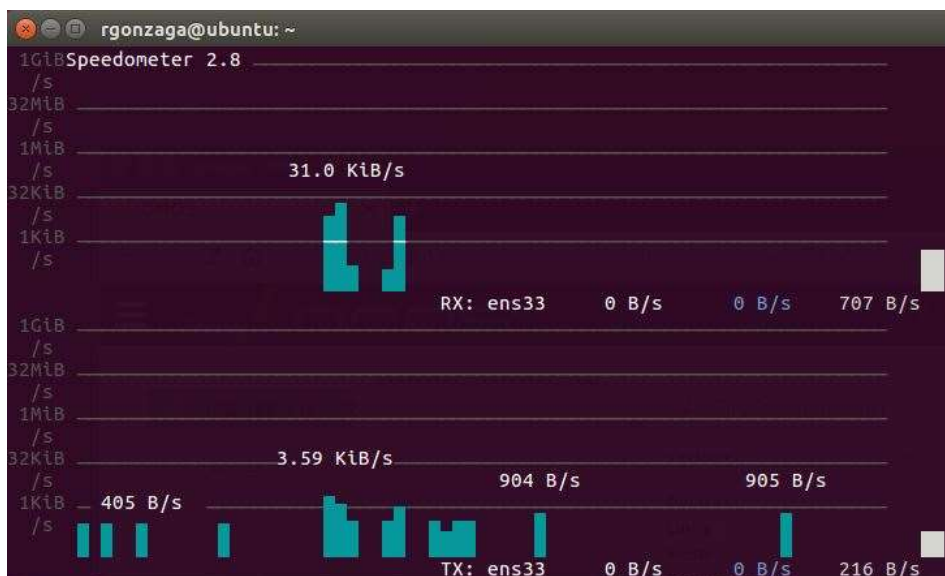
▶ Ethernet II, Src: Vmware_51:57:92 (08:0c:29:51:57:92), Dst: Vmware_42:e1:37 (08:0c:29:42:e1:37)
 ▶ Internet Protocol Version 4, Src: 29.18.171.166, Dst: 172.16.0.1
 ▶ Transmission Control Protocol, Src Port: 58975, Dst Port: 80, Seq: 0, Len: 0

Fonte: Elaboração própria do autor desta dissertação (2019).

Na coluna “*Source*” são identificados os endereçamentos de IPs falsos criados aleatoriamente pela ferramenta Hping3, onde cada endereço é responsável pelo envio de pacotes TCP SYN. O endereço de IP do Controlador SDN é apresentado na coluna “*Destination*”, uma vez que é o alvo do ataque. É possível ainda verificar o tempo de envio e a quantidade de pacotes enviados.

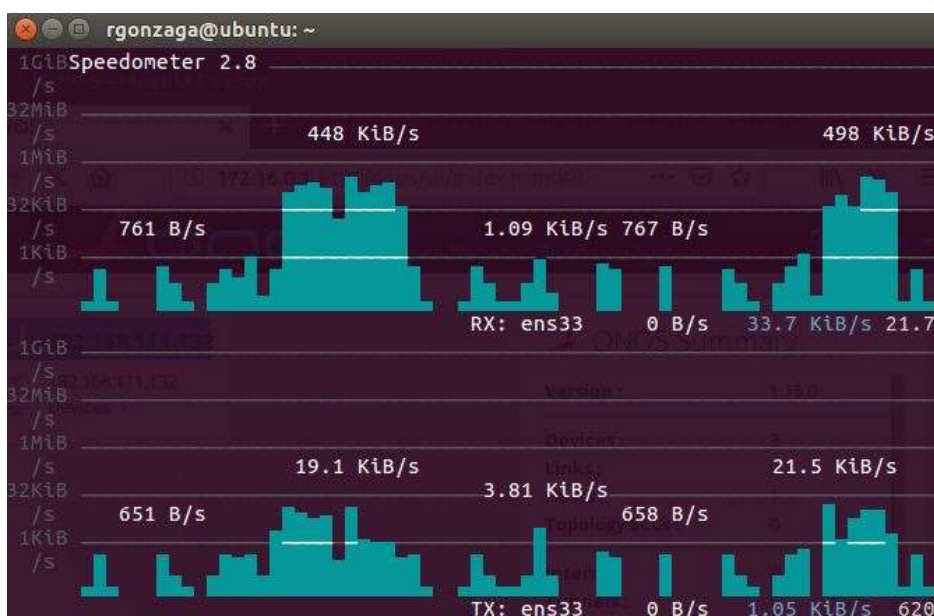
Os ataques DoS utilizados para os experimentos, foram executados em sua forma mais simples e ainda assim, com apenas alguns hosts, levou aproximadamente 15 (quinze) minutos para exaurir os recursos do Controlador SDN e torna-lo indisponível na rede. As consequências do ataque passaram por várias etapas antes de indisponibilizar recursos e serviços do controlador, como extrema lentidão e falhas sucessivas em aplicativos e processos. Através da ferramenta Speedometer foi possível monitorar o tráfego de entrada e saída da interface do controlador ONOS. A Figura 24 apresenta o gráfico do tráfego da interface do controlador antes do ataque, em quanto na Figura 25 é possível notar o crescente aumento no volume do tráfego gerado pelo ataque de SYN Flood.

Figura 24 – Captura do tráfego na interface de rede do controlador SDN antes do ataque de SYN Flood



Fonte: Elaboração própria do autor desta dissertação (2019).

Figura 25 – Captura do tráfego na interface de rede do controlador SDN no momento do ataque de SYN Flood



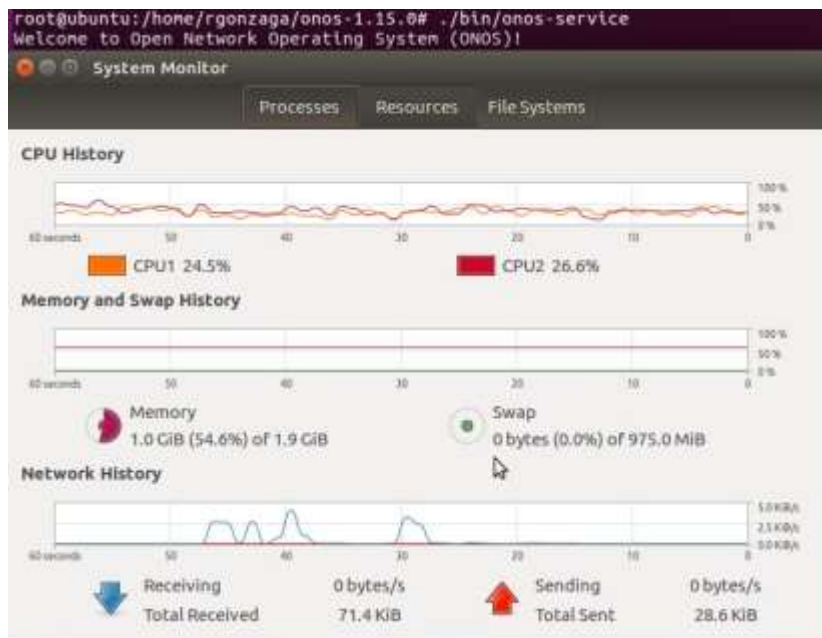
Fonte: Elaboração própria do autor desta dissertação (2019).

A parte superior do gráfico representa o fluxo de entrada de tráfego na interface de rede do controlador ONOS, enquanto a parte inferior apresenta o fluxo de saída. O fluxo de entrada representa o envio dos pacotes TCP SYN e a saída os pacotes de ACK enviados em resposta a solicitação de conexão requisitada pelos pacotes do ataque.

Além da sobrecarga na interface de rede, o processamento do controlador ONOS foi seriamente comprometida, afetando diretamente o desempenho do sistema, ocasionando extrema lentidão e falha em serviços e aplicativos do controlador. As Figuras 26 e 27

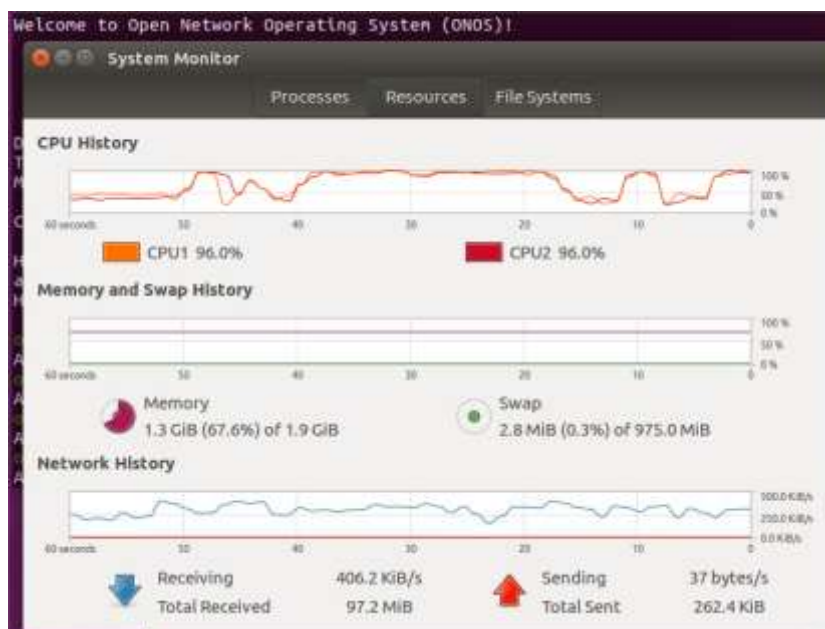
apresentam o monitoramento do sistema antes e depois da realização dos ataques e comprometimento dos recursos do controlador.

Figura 26 – Monitoramento de recursos do Controlador ONOS antes do ataque de SYN Flood



Fonte: Elaboração própria do autor desta dissertação (2019).

Figura 27 – Monitoramento de recursos do Controlador ONOS durante o ataque de SYN Flood



Fonte: Elaboração própria do autor desta dissertação (2019).

O alto consumo de processamento e dos recursos de rede, ocasionou diversos erros na identificação da topologia (*links*, fluxos, dispositivos, endereçamentos IPs e aplicativos) durante o ataque, antes do software controlador ficar totalmente indisponível. O aplicativo de

encaminhamento FWD apresentou travamento e falhas no processo de gerenciamento dinâmico de fluxo como pode ser identificado na Figura 28.

Figura 28 – Falha de gerenciamento dos fluxos pelo aplicativo FWD do ONOS



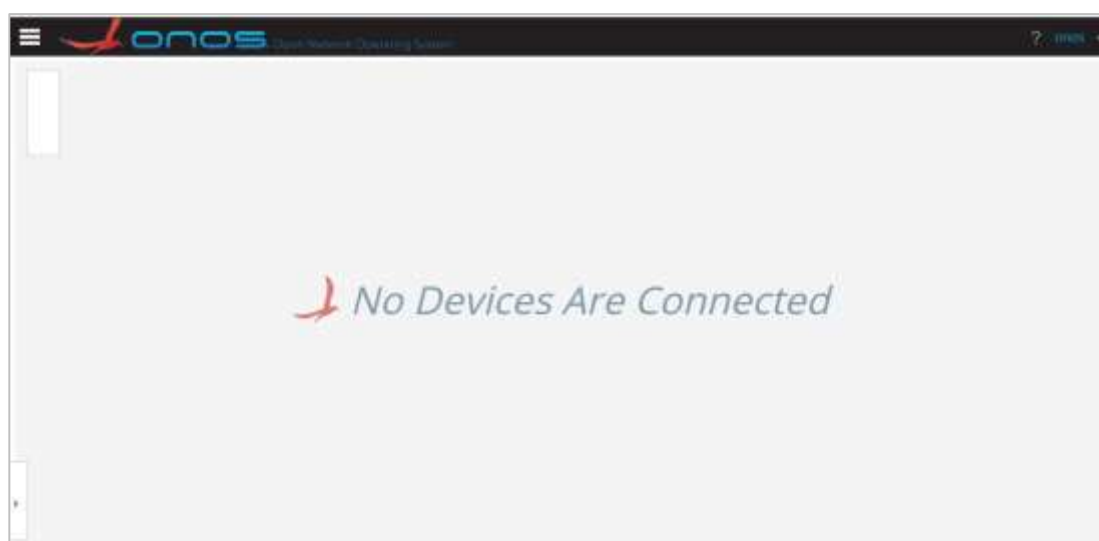
STATE	FACETS	DURATION	FLOW PRIORITY	TABLE NAME	SELECTOR	TREATMENT	APP NAME
Active	558	1.307	40000	0	ETH_TYPE:isip	in@[OUTPUTCONTROLLER] clear=true	core
Active	0	65	40000	0	ETH_TYPE:ip	in@[OUTPUTCONTROLLER] clear=true	core
Active	32.207	1.307	5	0	ETH_TYPE:ip	in@[OUTPUTCONTROLLER] clear=true	core
Active	582	1.307	40000	0	ETH_TYPE:isip	in@[OUTPUTCONTROLLER] clear=true	core

Fonte: Elaboração própria do autor desta dissertação (2019).

As falhas no aplicativo de encaminhamento afetaram diretamente o gerenciamento de fluxo do controlador, impossibilitando o ONOS de acompanhar as requisições, adicionar e contabilizar corretamente as entradas de fluxo válidas na tabela do controlador. Esse processo ocasionou falhas na interação entre o controlador e o *switch OpenFlow*, desencadeando uma degradação na comunicação da camada de controle com a camada de dados.

O último efeito do ataque foi total indisponibilidade dos serviços do controlador ONOS, que deixou de funcionar e indisponibilizou toda a camada de controle, uma vez que os serviços básicos do controlador se tornaram inacessíveis como pode ser visto nas Figuras 29 e 30.

Figura 29 – Falha do Controlador ONOS na identificação da topologia da rede



Fonte: Elaboração própria do autor desta dissertação (2019).

Figura 30 – Falha total na GUI Web do controlador ONOS



Fonte: Elaboração própria do autor desta dissertação (2019).

Por ser baseado na tecnologia JAVA, vários recursos do ONOS dependem de bibliotecas e frameworks para funcionar corretamente, como é o exemplo Jetty, software cliente-servidor utilizado pelo ONOS. O erro apresentado na Figura 16 demonstram que os ataques de SYN Flood e ICMP Flood indisponibilizaram toda a estrutura da GUI do ONOS, tirando o serviço Jetty de operação.

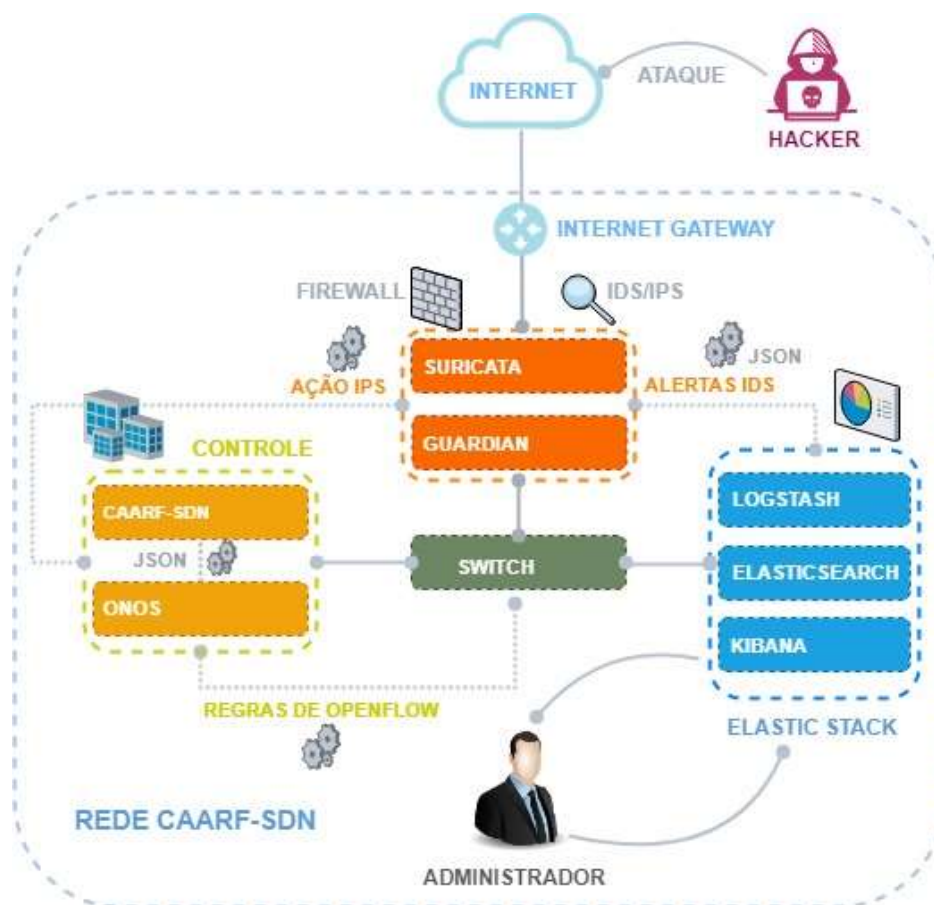
Após os ataques realizados e as falhas apresentadas, a camada de controle da rede ficou indisponível acarretando problemas em toda a rede, uma vez que vários serviços como a implementação e monitoramento de fluxo deixam de ser aplicados. Nesse cenário, a aplicação do arcabouço CAARF-SDN torna-se inviável, uma vez que a indisponibilidade do controlador, ocasiona falha no processo de reprogramação automática da rede, que é caracterizada como principal proposta do projeto CAAR-SDN.

5.5 CENÁRIO 2 – TOLERÂNCIA A FALHAS NO CONTEXTO CAARF-SDN

Tendo em vista as vulnerabilidades apresentadas nos experimentos anteriores, o objetivo desse cenário é a implementação de uma solução de segurança, visando a proteção dos ativos críticos da rede e mitigar, riscos e prevenir a ocorrência das falhas identificadas no ambiente de rede do arcabouço CAARF-SDN.

Basicamente a solução proposta para esse cenário é baseada em mecanismos IDS/IPS, visando a proteção da rede contra-ataques externos. Os sistemas de IDS foram apresentados no capítulo 4 como possíveis propostas de soluções para mitigação de ataques DoS e DDoS em redes baseadas no paradigma SDN. É ilustrada na Figura 31 um modelo conceitual da arquitetura geral proposto para esse cenário de tolerância a falhas.

Figura 31 – Arquitetura conceitual do cenário de tolerância a falhas



Fonte: Elaboração própria do autor desta dissertação (2019).

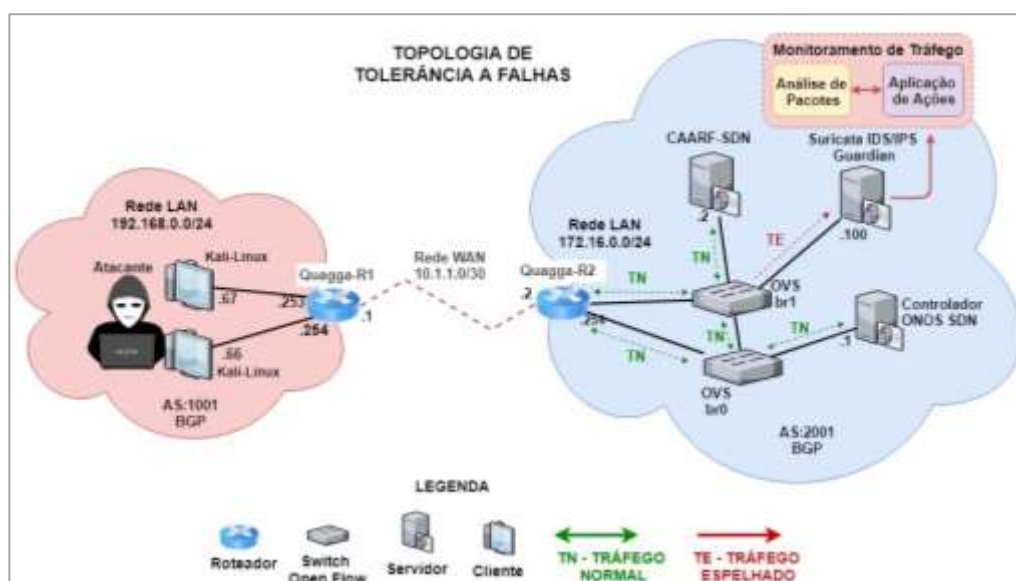
Essa arquitetura é detalhada no decorrer dos experimentos dos cenários, apresentando o funcionamento dos mecanismos no processo de identificação e contenção de ameaças no contexto de rede do arcabouço CAARF-SDN.

5.5.1 Solução de Tolerância a Falhas na rede CAARF-SDN

A solução de segurança proposta para esse cenário, é a implementação e configuração personalizada de um conjunto de mecanismos de segurança visando mitigar ameaças relacionadas a ataques DoS no ambiente de rede do arcabouço CAARF-SDN. A ferramenta Suricata foi implementada como IDS para monitoramento, coleta e análise de todos os pacotes direcionados a rede do CAAR-SDN, através da configuração de assinaturas de comportamento de tráfego, visando a identificação e bloqueio de ameaças de forma autônoma. A implementação de um IDS em ambientes SDN, como o proposto no cenário do CAARF-SDN podem ocorrer de duas formas. Na primeira, o IDS pode ser implementado de forma menos

intrusiva, onde todo o tráfego de entrada da rede pode ser espelhado direto para a porta do dispositivo IDS, através de recursos de espelhamento existente nos próprios encaminhadores *OpenFlow* como é o caso do *OpenVSwitch*. O tráfego espelhado é monitorado e em caso de detecção de tráfego malicioso, alertas são gerados e o bloqueio do tráfego é solicitado ao dispositivo IPS, que envia políticas de bloqueio para uma aplicação de gerenciamento de políticas de firewall do controlador ONOS através da API REST. A Figura 32 ilustra um modelo conceitual do cenário dessa forma de implementação.

Figura 32 – Topologia conceitual de implementação de IDS para tráfego espelhado



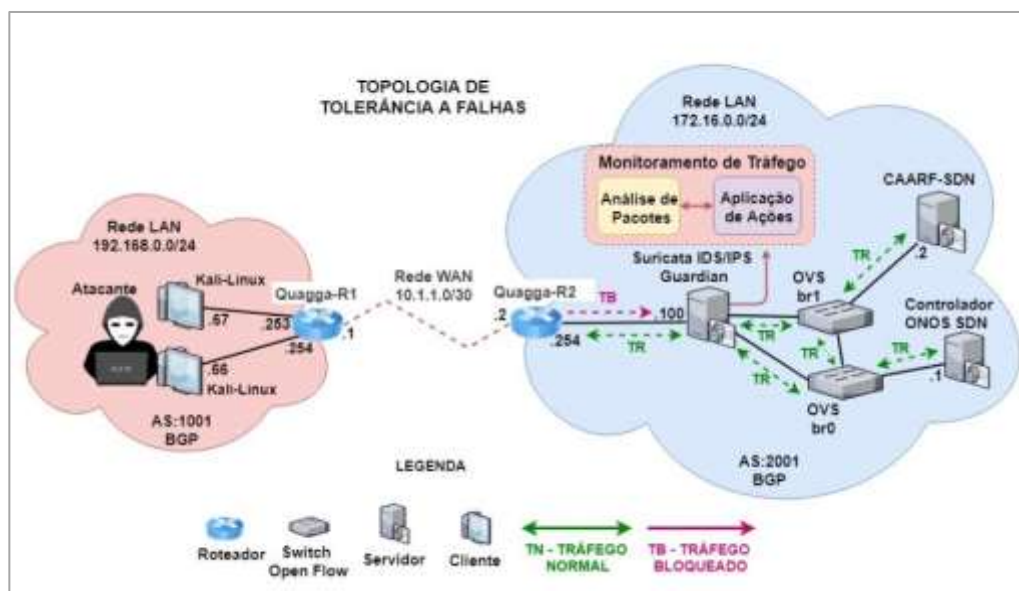
Fonte: Elaboração própria do autor desta dissertação (2019).

O *software* de controle ONOS dispõe de uma aplicação denominada de *Policy Framework For ONOS*, que proporciona a interação com sistemas IPS através de arquivos JSON, que informam a aplicação sobre parâmetros do tráfego malicioso identificado e solicita o bloqueio (WIKI ONOS, 2019). Esse é um método efetivo, entretanto nesse tipo de abordagem, é necessário o desenvolvimento de aplicações para gerar arquivos de *log* em formato “json” contendo exatamente todos os parâmetros especificados pela aplicação, o que dificulta sua implementação devido ao grau de complexidade e a falta de documentação mais detalhada e clara sobre a ferramenta. Sendo assim, a apresentação desse formato tem por objetivo a motivação e justificativa futuras pesquisas baseados nesse trabalho, tendo em vista que explora recursos fundamentais da arquitetura SDN.

A segunda forma de implementação é mais convencional e foi utilizada na presente dissertação para validar a eficiência e a importância dos mecanismos de segurança no contexto das redes híbridas, em específico o cenário do CAARF-SDN. Nessa proposta, a ferramenta Suricata é implementada como um NIDS *inline*, coletando todos o tráfego da rede que passa

diretamente por sua interface. O modelo conceitual dessa abordagem de implementação é apresentado na Figura 33.

Figura 33 – Topologia conceitual de implementação de IDS *inline* no ambiente de rede do arcabouço CAARF-SDN




Fonte: Elaboração própria do autor desta dissertação (2019).

Todo o monitoramento e a análise do tráfego da rede ocorrem em tempo real, efetuando comparações de anomalias identificadas com o a base de dados de assinaturas de comportamento de tráfego. Ao identificar um comportamento suspeito em determinado tráfego, o Suricata gera um *log* com alertas e solicita a ferramenta Guardian, o bloqueio do tráfego suspeito. O Guardian recebe o alerta e executa regras de bloqueio no *firewall* do host NIDS, interrompendo a conexão suspeita. Além do Suricata, as ferramentas Kibana, Elastic e Logstash foram utilizadas para viabilizar uma interface gráfica mais intuitiva para apresentação dos resultados em tempo real. Por fim, a aplicação Guardian foi implementada e configurada para realizar ações de bloqueio de ameaças, baseadas nos *logs* de monitoramento do Suricata.

Os alertas de intrusão foram devidamente configurados, baseados na assinatura do comportamento de ataques DoS do tipo SYN Flood e ICMP Flood, com o objetivo de analisar o tráfego suspeito e gerar alertas de intrusão. A Figura 34 apresenta as configurações de assinatura que disparam os sensores do Suricata sobre o comportamento malicioso no tráfego da rede.

Figura 34 – Configuração das regras de alertas maliciosos do Suricata



```
root@ubuntu: /etc/init.d

alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"Possible
TCP Syn Flood Attck DoS port 80"; flags:S; flow:to_server; thresho
ld: type both, track by_src, count 40, seconds 5; classtype:attempt
ed-dos; sid:10001; rev:1;)

alert icmp any any -> any any (msg:"Possible Smurf Attack"; sid:100
0003; itype:8;)
```

Fonte: Elaboração própria do autor desta dissertação (2019).

A primeira regra, faz a análise do tráfego TCP de origem externa a rede CAARF-SDN, buscando a identificação de anomalias no comportamento do tráfego, que indiquem possíveis ameaças de intrusão. Essa regra faz a comparação do volume de pacotes do tipo TCP SYN são enviados para a rede em um dado intervalo de tempo através da porta 80. Nesse caso a assinatura foi configurada para gerar alertas de ameaça caso o envio de requisições SYN passem de 40 pacotes a cada 5 segundos. A segunda regra faz a análise de tráfegos do tipo ICMP, identificando anormalidades no volume de pacotes do tipo *Echo* ICMP, gerando alertas toda vez que ocorrer a violação das regras estabelecidas no Suricata. Diferente da primeira regra, a assinatura de comportamento de ICMP, faz a análise mais ampla do tráfego da rede, uma vez que algumas técnicas de ICMP Flood podem gerar um grande volume de *Echo Response* entre dispositivos na própria rede local. É importante destacar que além da criação do arquivo de regras “caarf.rules”, outras configurações foram realizadas e serão descritas no anexo da presente dissertação. Entretanto os parâmetros mais importantes para configuração do Suricata no ambiente de rede do arcabouço CAARF-SDN são apresentados na Figura 35.

Figura 35 – Script do Suricata para configuração e identificação das redes

```

root@ubuntu: /home/rgonzaga
vars:
# more specific is better for alert accuracy and performance
address-groups:
HOME_NET: "[172.16.0.0/24]"
#HOME_NET: "[192.168.0.0/16]"
#HOME_NET: "[10.0.0.0/8]"
#HOME_NET: "[172.16.0.0/12]"
#HOME_NET: "any"

EXTERNAL_NET: "!$HOME_NET"
#EXTERNAL_NET: "any"

HTTP_SERVERS: "$HOME_NET"
SMTP_SERVERS: "$HOME_NET"
SQL_SERVERS: "$HOME_NET"
DNS_SERVERS: "$HOME_NET"
TELNET_SERVERS: "$HOME_NET"
AIM_SERVERS: "$EXTERNAL_NET"
DNP3_SERVER: "$HOME_NET"
DNP3_CLIENT: "$HOME_NET"
MODBUS_CLIENT: "$HOME_NET"
MODBUS_SERVER: "$HOME_NET"
ENIP_CLIENT: "$HOME_NET"
ENIP_SERVER: "$HOME_NET"

port-groups:
HTTP_PORTS: "80"

```

Fonte: Elaboração própria do autor desta dissertação (2019).

Variáveis importantes como “*\$HOME_NET*” e “*\$EXTERNAL_NET*” são configuradas nesse arquivo, para identificar os endereços de IP da rede local do CAARF-SDN, assim como para identificar todos os endereços que representam tráfego externo. Variáveis de identificação de serviços e portas também são configuradas com o objetivo de viabilizar um monitoramento mais profundo e eficiente. Na sessão de regras são inseridos os nomes dos arquivos com as regras de alerta, assim como o local de armazenamento desses arquivos, como é apresentado na Figura 36.

Figura 36 – Script de configuração da localização das regras do Suricata

```

##
## Step 2: select the rules to enable or disable
##
default-rule-path: /etc/suricata/rules
rule-files:
- caarf.rules
- botcc.rules
# - botcc.portgrouped.rules
# - ciarmy.rules

```

Fonte: Elaboração própria do autor desta dissertação (2019).

Para esse experimento o arquivo “caarf.rules”, apresentado detalhadamente na Figura 41, foi criado com todas as regras ou assinaturas de comportamento de tráfego malicioso que são utilizadas para geração de alertas dos tráfegos maliciosos.

Em complemento ao serviço de monitoramento e alertas do Suricata, a ferramenta Guardian é proposta e implementada como uma solução poderosa para reação a ataques. Sua estrutura é formada por diversos arquivos que foram configurados para atender as necessidades do projeto CAARF-SDN. Entre eles os principais são o módulo de bloqueio “*guardian_block.sh*” e o módulo de desbloqueio “*guardian_unblock.sh*”, pelos quais são executados o bloqueio e desbloqueio de tráfego de forma automatizada, baseado nos *logs* do Suricata. As demais configurações referentes a interfaces de redes, endereçamentos IPs e caminhos de arquivos de *log* são encontrados no arquivo raiz das configurações denominado “*guardian.conf*”. As configurações adequadas para habilitar a execução do guardian no ambiente de rede do arcabouço CAARF-SDN podem ser vista no arquivo raiz do Guardian apresentados na Figura 37.

Figura 37 – Script de configuração da ferramenta Guardian

```

root@ubuntu: /home/rgonzaga
# The machines IP address that is visable to the internet
# If this is left undefined, then guardian will attempt to get the information
# from ifconfig, as long as it has an interface to use. This would be useful
# for people on ppp links, or dhcp machines, or if you are lazy :)
HostIpAddr 10.1.1.2

# The IP address of remote controller which will be requested to
# block/unblock hosts
RemoteController 172.16.0.1

# Here we define the interface which we will use to guess the IP address, and
# block incoming offending packets. This is the only option that is required
# for guardian to run. If the rest are undefined, guardian will use the default.
Interface ens43

# The last octet of the ip address, which gives us the gateway address.
HostGatewayByte 1

# Guardian's log file
LogFile /var/log/guardian.log

# Snort's alert file. This can be the snort.alert file, or a syslog file
# There might be some snort alerts that get logged to syslog which guardian
# might not see..
AlertFile /var/log/suricata/fast.log

```

Fonte: Cooper (2019).

O arquivo indica para o Guardian em qual endereço de IP e *interface* os bloqueios ou desbloqueios devem ser realizados, através dos parâmetros “HostIpAddr” e *interface*, que no experimento foram configurados apontando para o roteador Quagga R2, por ser a interface

WAN por onde o tráfego externo entra na rede e todos os bloqueios são realizados nessa interface, a mesma que está sendo monitorada pelo Suricata. Na variável “*RemoteController*” é apontado o host do controlador da rede para casos de interações remotas. Por fim é indicado a localização do arquivo com os alertas gerados pela ferramenta Suricata.

5.5.2 Contenção de Ataques SYN Flood na Rede do Arcabouço CAARF-SDN

Nesse cenário, os mesmos ataques descritos na sessão 5.4.1, foram executados com o objetivo de esgotar os recursos e ocasionar falhas e indisponibilidade de serviços. Os ataques DoS direcionados para o Controlador SDN pode ser visto na Figura 38, assim como no arcabouço CAARF-SDN é apresentado na Figura 39.

Figura 38 – Ataque SYN Flood direcionado ao Controlador ONOS

```
root@kali-sdn:~#  
root@kali-sdn:~# hping3 --fast -S -p 80 172.16.0.1  
HPING 172.16.0.1 (eth0 172.16.0.1): S set, 40 headers + 0 data bytes
```

Fonte: Elaboração própria do autor desta dissertação (2019).

Figura 39 – Ataque SYN Flood direcionado ao hospedeiro de referência do arcabouço CAARF-SDN

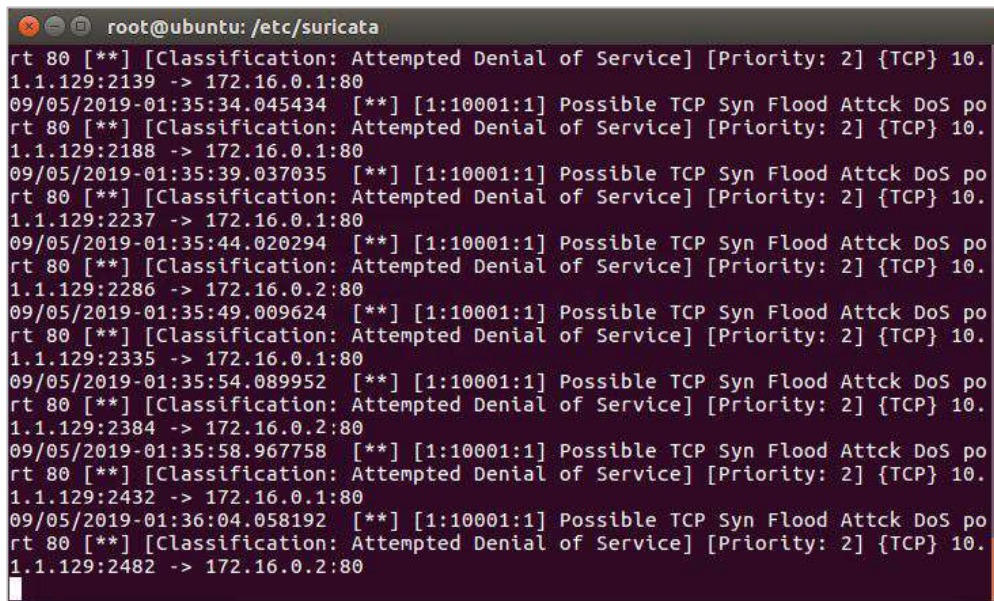
```
root@kali-sdn:~#  
root@kali-sdn:~# hping3 --fast -S -p 80 172.16.0.2  
HPING 172.16.0.2 (eth0 172.16.0.2): S set, 40 headers + 0 data bytes
```

Fonte: Elaboração própria do autor desta dissertação (2019).

Ambos os ataques foram executados com o mesmo objetivo apresentados na sessão 5.4.1, de inundar a interface do Controlador SDN e do arcabouço CAARF-SDN requisições TCP SYN, esgotando a capacidade de manter os serviços de rede operacionais. Entretanto o presente cenário se diferencia devido aos mecanismos de IDS/IPS implementados para tratar as tentativas de intrusão e ameaças identificadas, buscando provisionar maior confiabilidade e tolerância a falhas ocasionadas por ameaças no ambiente de rede.

Logo no início do ataque, a ferramenta Suricata detecta em tempo real as tentativas de inundação de pacotes TCP SYN, e gera diversos alertas relatando as ocorrências. Esse processo de geração de alerta em tempo real e registro no arquivo de log pode ser verificado na Figura 40.

Figura 40 – Alertas dos ataques SYN Flood identificados pelo Suricata IDS



```

root@ubuntu: /etc/suricata
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2139 -> 172.16.0.1:80
09/05/2019-01:35:34.045434 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2188 -> 172.16.0.1:80
09/05/2019-01:35:39.037035 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2237 -> 172.16.0.1:80
09/05/2019-01:35:44.020294 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2286 -> 172.16.0.2:80
09/05/2019-01:35:49.009624 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2335 -> 172.16.0.1:80
09/05/2019-01:35:54.089952 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2384 -> 172.16.0.2:80
09/05/2019-01:35:58.967758 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2432 -> 172.16.0.1:80
09/05/2019-01:36:04.058192 [**] [1:10001:1] Possible TCP Syn Flood Attck DoS po
rt 80 [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 10.1.1.129:2482 -> 172.16.0.2:80

```

Fonte: Elaboração própria do autor desta dissertação (2019).

No *log* de alerta gerado pelo Suricata é possível identificar informações de extrema importância, como a possível origem do ataque, hospedeiro alvo e a mensagem personalizada que ajuda na identificação do tipo de ataque. Após identificação do ataque, o *log* do Suricata é utilizado para executar rapidamente as políticas de contenção do ataque em andamento. A ferramenta responsável pelo bloqueio do ataque é o Guardian como já foi abordado em sessões anteriores. Ao receber os alertas do Suricata sobre o comportamento malicioso no tráfego, o Guardian executa as regras necessárias para ativar o bloqueio do fluxo intrusivo no *firewall* da rede, que nesse caso específico é o próprio hospedeiro IDS/IPS. A reação aos comportamentos maliciosos nos tráfegos monitorados da rede CAARF-SDN são executadas em tempo real e de forma efetiva, ocasionando o bloqueio do ataque realizado em alguns segundos como pode ser visto na Figura 41.

Figura 41 - Perda de pacotes TCP após a contenção do ataque

```

root@kali-sdn: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
ms
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=68 win=29200 rtt=7.0 m
s
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=69 win=29200 rtt=4.4 m
s
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=70 win=29200 rtt=34.3
ms
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=71 win=29200 rtt=3.0 m
s
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=72 win=29200 rtt=2.6 m
s
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=73 win=29200 rtt=4.3 m
s
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=74 win=29200 rtt=16.2
ms
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=75 win=29200 rtt=13.9
ms
len=46 ip=172.16.0.1 ttl=64 DF id=0 sport=80 flags=SA seq=76 win=29200 rtt=5.5 m
s
^C
--- 172.16.0.1 hping statistic ---
1061 packets transmitted, 77 packets received, 93% packet loss
round-trip min/avg/max = 2.1/9.7/147.5 ms

```

Fonte: Elaboração própria do autor desta dissertação (2019).

É possível notar que o hospedeiro atacante perde a conexão e não consegue enviar pacotes TCP SYN e deixa de receber pacotes SYN ACK devido ao bloqueio efetuado pela ferramenta Guardian. É notório o grande volume de pacotes descartados antes do tráfego ser totalmente interrompido e ainda o tempo que levou para o tráfego ser interrompido.

Nas análises realizadas através da ferramenta Wireshark é possível ainda verificar o comportamento do ataque antes da implementação do mecanismo de segurança ser implementado, como é apresentado nas Figura 42.

Figura 42 – Análise do ataque SYN Flood no Controlador SDN através da ferramenta Wirehsark

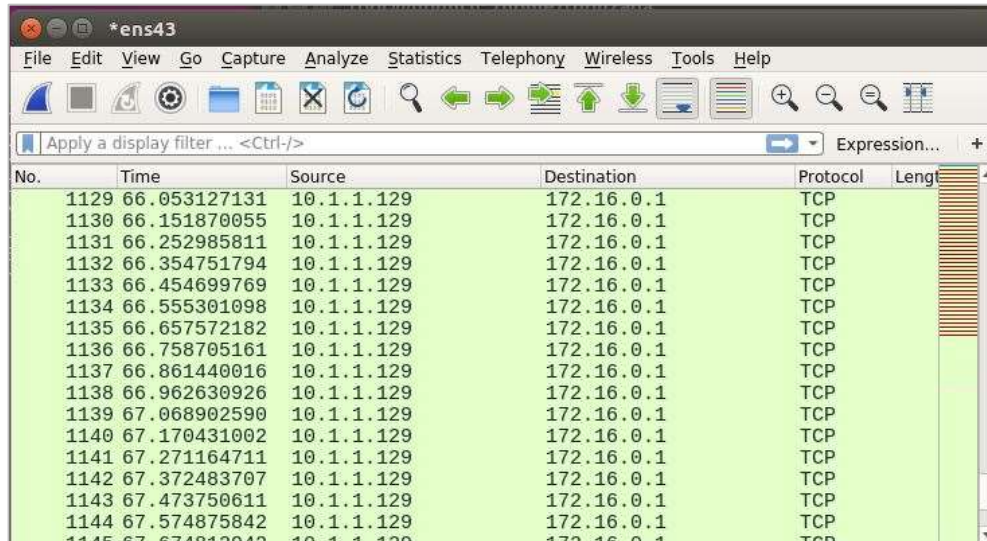
No.	Time	Source	Destination	Protocol	Length
727	44.211295866	172.16.0.1	10.1.1.129	TCP	
728	44.219373454	10.1.1.129	172.16.0.1	TCP	
729	44.296519730	10.1.1.129	172.16.0.1	TCP	
730	44.296696701	172.16.0.1	10.1.1.129	TCP	
731	44.298028815	10.1.1.129	172.16.0.1	TCP	
732	44.398645281	10.1.1.129	172.16.0.1	TCP	
733	44.398841206	172.16.0.1	10.1.1.129	TCP	
734	44.401213343	10.1.1.129	172.16.0.1	TCP	
735	44.498367509	10.1.1.129	172.16.0.1	TCP	
736	44.498560576	172.16.0.1	10.1.1.129	TCP	

Fonte: Elaboração própria do autor desta dissertação (2019).

É possível notar o envio massivo de requisições TCP (SYN) efetuadas pelo IP (10.1.1.1) do roteador R1 da rede atacante para o IP (172.16.0.1) do hospedeiro de referência do

arcabouço CAARF-SDN, assim como os pacotes TCP (SYN ACK) enviados em resposta. Em contraste, a Figura 43 apresenta a análise do tráfego no momento em que os mecanismos de segurança fazem a identificação e contenção do ataque.

Figura 43 – Análise da contenção do ataque SYN Flood no Controlador ONOS através da ferramenta Wireshark

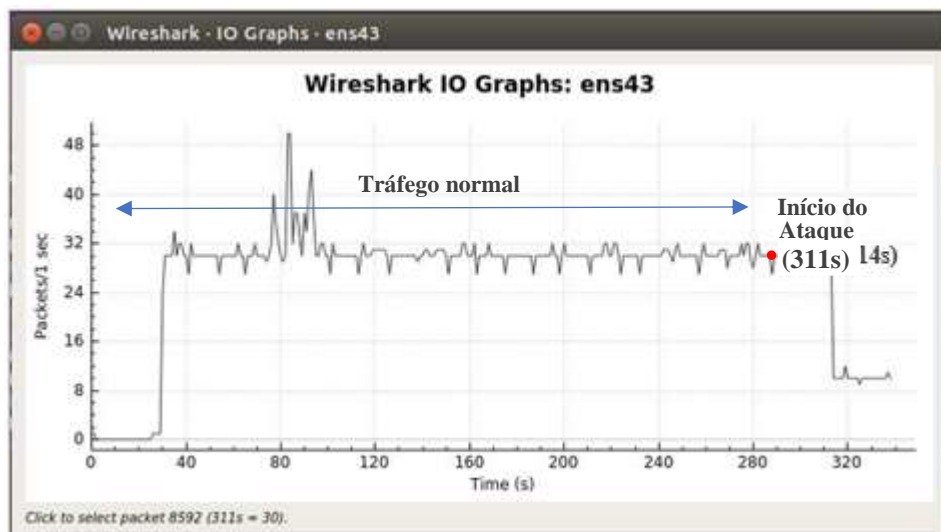


No.	Time	Source	Destination	Protocol	Length
1129	66.053127131	10.1.1.129	172.16.0.1	TCP	
1130	66.151870055	10.1.1.129	172.16.0.1	TCP	
1131	66.252985811	10.1.1.129	172.16.0.1	TCP	
1132	66.354751794	10.1.1.129	172.16.0.1	TCP	
1133	66.454699769	10.1.1.129	172.16.0.1	TCP	
1134	66.555301098	10.1.1.129	172.16.0.1	TCP	
1135	66.657572182	10.1.1.129	172.16.0.1	TCP	
1136	66.758705161	10.1.1.129	172.16.0.1	TCP	
1137	66.861440016	10.1.1.129	172.16.0.1	TCP	
1138	66.962630926	10.1.1.129	172.16.0.1	TCP	
1139	67.068902590	10.1.1.129	172.16.0.1	TCP	
1140	67.170431002	10.1.1.129	172.16.0.1	TCP	
1141	67.271164711	10.1.1.129	172.16.0.1	TCP	
1142	67.372483707	10.1.1.129	172.16.0.1	TCP	
1143	67.473750611	10.1.1.129	172.16.0.1	TCP	
1144	67.574875842	10.1.1.129	172.16.0.1	TCP	
1145	67.674812042	10.1.1.129	172.16.0.1	TCP	

Fonte: Elaboração própria do autor desta dissertação (2019).

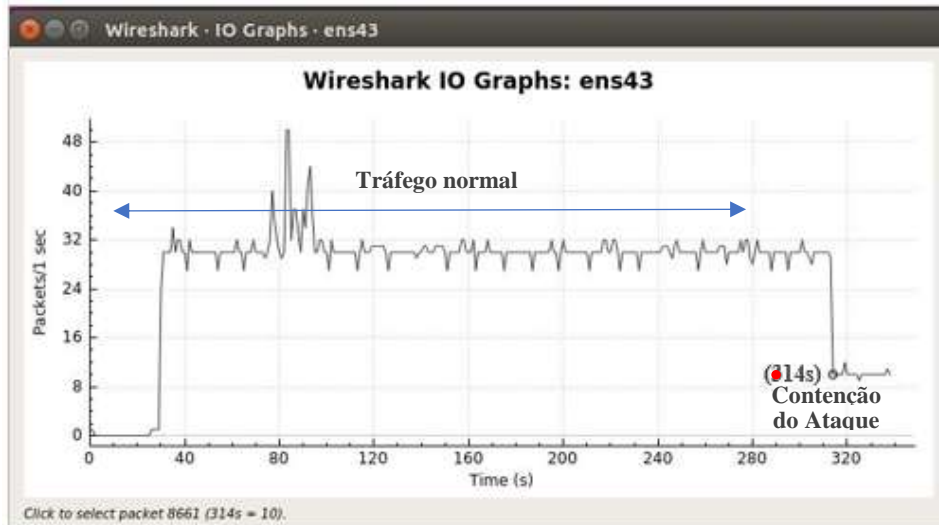
Após o bloqueio, o host atacante deixa de ter acesso a conexão do host do arcabouço CAARF-SDN, encerrando a transação TCP. Dessa forma o arcabouço deixa de receber solicitações SYN e de responder com SYN ACK, evitando a alocação de recursos para conexões falsas requisições TCP. Os Gráfico 1 e 2 demonstram de forma mais clara o momento exato do início e término do ataque evidenciando a efetividade dos mecanismos na contenção do ataque.

Gráfico 1– Análise de gráfico Wireshark da execução do ataque SYN Flood no Controlador ONOS no momento do início da contenção do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

Gráfico 2 – Análise de gráfico Wirshark da execução do ataque SYN Flood no Controlador ONOS e do momento exato do bloqueio do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

Os picos no tráfego representam o aumento do envio e recebimento de pacotes TCP durante o período do ataque. Entretanto esse pico começa a cair indicando o início do bloqueio realizado pelos mecanismos de segurança aplicados. A reação ao ataque levou exatamente 3 segundos, sendo que o ataque teve seu início aos 311 segundos da execução do experimento e o seu bloqueio aos 314 segundo como pode ser visto nos Gráficos 1 e 2.

5.5.3 Contenção de Ataques ICMP Flood no Contexto do CAARF-SDN

Nesse segundo ataque, foi executado um tipo específico de ICMP Flood denominado de *Smurf*, apresentado na sessão 5.4.1. Esse ataque teve por objetivo redirecionar diversos pacotes de ICMP *Replay* para o Controlador SDN e arcabouço CAARF-SDN, na tentativa de sobrecarregar a interface desses dispositivos e indisponibilizar seus recursos. A Figura 44 apresenta os parâmetros do ataque sendo executados no host atacante tendo como alvo o hospedeiro de referência do Controlador SDN e a Figura 45 apresenta o mesmo ataque sendo direcionado para o hospedeiro de referência do arcabouço CAARF-SDN.

Figura 44 – Ataque ICMP Flood *Smurf* direcionado para o hospedeiro de referência do arcabouço CAARF-SDN

```

root@kali-sdn: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali-sdn:~# hping3 --icmp --flood -c 1000 --spooof 172.16.0.1 172.16.0.255
HPING 172.16.0.255 (eth0 172.16.0.255): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown

```

Fonte: Elaboração própria do autor desta dissertação (2019).

Figura 45 – Ataque ICMP Flood *Smurf* direcionado para Controlador ONOS

```

root@kali-sdn: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali-sdn:~# hping3 --icmp --flood -c 1000 --spooof 172.16.0.2 172.16.0.255
HPING 172.16.0.255 (eth0 172.16.0.255): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown

```

No.	Time	Source	Destination	Protocol
3189	506.127978325	172.16.0.1	172.16.0.255	ICMP

Fonte: Elaboração própria do autor desta dissertação (2019).

Em ambos os comandos, os endereços de IPs dos hospedeiros alvos são intencionalmente mascarados como endereço de *broadcast* (172.16.0.255) da rede, com a finalidade de enviar pacotes de *Echo Request* para todos os dispositivos ativos que por sua vez irão responder as solicitações com pacotes de *Echo Replay*. Na Figura 46 é possível verificar detalhes do ataque *Smurf* executado na rede representativa do arcabouço CAARF-SDN.

Figura 46 – Análise Wireshark do ataque ICMP Flood *Smurf* direcionado a rede do CAARF-SDN

No.	Time	Source	Destination	Protocol	Length
2251	269.392150169	172.16.0.143	172.16.0.1	DNS	
2252	269.392208557	172.16.0.143	172.16.0.1	DNS	
2253	271.180669847	172.16.0.140	172.16.0.1	DNS	
2254	271.180735987	172.16.0.1	172.16.0.140	ICMP	1
2255	271.180945303	172.16.0.140	172.16.0.1	DNS	
2256	271.180998942	172.16.0.1	172.16.0.140	ICMP	1
2257	271.182402345	172.16.0.140	172.16.0.1	DNS	
2258	271.182490905	172.16.0.1	172.16.0.140	ICMP	1
2259	271.182679338	172.16.0.140	172.16.0.1	DNS	
2260	271.182726761	172.16.0.1	172.16.0.140	ICMP	1
2261	271.185365668	172.16.0.140	172.16.0.1	DNS	
2262	271.185421332	172.16.0.1	172.16.0.140	ICMP	1
2263	271.185601524	172.16.0.140	172.16.0.1	DNS	
2264	271.187939133	172.16.0.140	172.16.0.1	DNS	
2265	271.187993470	172.16.0.140	172.16.0.1	DNS	
2266	272.307871145	172.16.0.139	172.16.0.1	DNS	
2267	272.307940009	172.16.0.1	172.16.0.139	ICMP	1
2268	272.308415073	172.16.0.139	172.16.0.1	DNS	

Fonte: Elaboração própria do autor desta dissertação (2019).

Várias requisições *Echo Request* são executadas pelo atacante que faz as requisições se passando pelo arcabouço CAARF-SDN identificado pelo IP 172.16.0.1 e o Controlador SDN pelo IP 172.16.0.2, levando os outros hosts da rede a responderem os *Echo Replay* diretamente para os servidores. Esse processo ocorre em *loop* consumindo recursos de processamento e sobrecarregando a *interface* de rede dos hospedeiros alvos. Entretanto, nesse cenário de tolerância a falhas os mecanismos de contenção de ameaças identificaram o ataque logo no

início de sua execução. A Figura 47 apresenta os alertas gerados pelo Suricata ao identificar o ataque de *Smurf*.

Figura 47 – Alertas de ataque de *Smurf* gerados pelo Suricata IDS

```

rgonzaga@ubuntu: ~
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.734996 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.735037 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.739746 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.739793 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.739828 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.741137 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.741176 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.741210 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.742279 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.742391 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0
09/05/2019-01:16:53.742430 [**] [1:1000003:0] Possible Smurf Attack [**] [Classi
fication: (null)] [Priority: 3] {ICMP} 172.16.0.1:8 -> 172.16.0.255:0

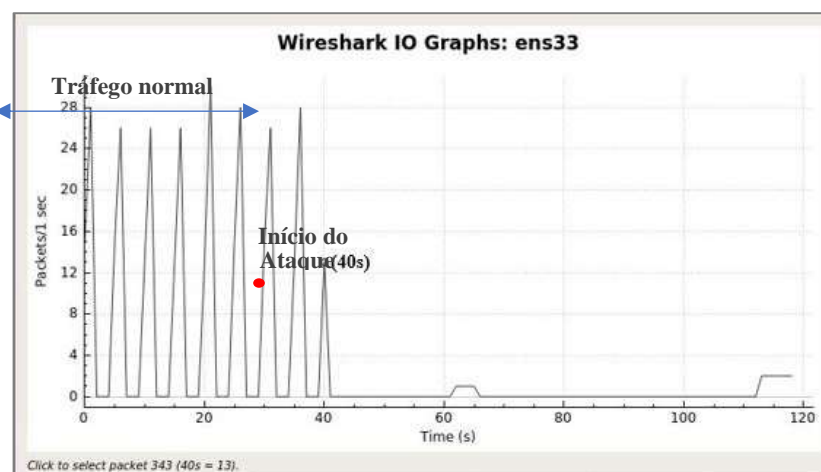
```

Fonte: Elaboração própria do autor desta dissertação (2019).

Assim como nos logs de análise do Wireshark é possível identificar nos alertas do *Suricata*, o envio de pacotes ICMP do tipo *Echo Requeste* em *loop* para o endereço *broadcast* da rede (172.16.0.255), ocasionando o repasse dos pacotes para todos os dispositivos da rede, que por sua vez retornam *Echo Replay* para o IP do Controlador SDN e o arcabouço CAARF-SDN como foi apresentado anteriormente.

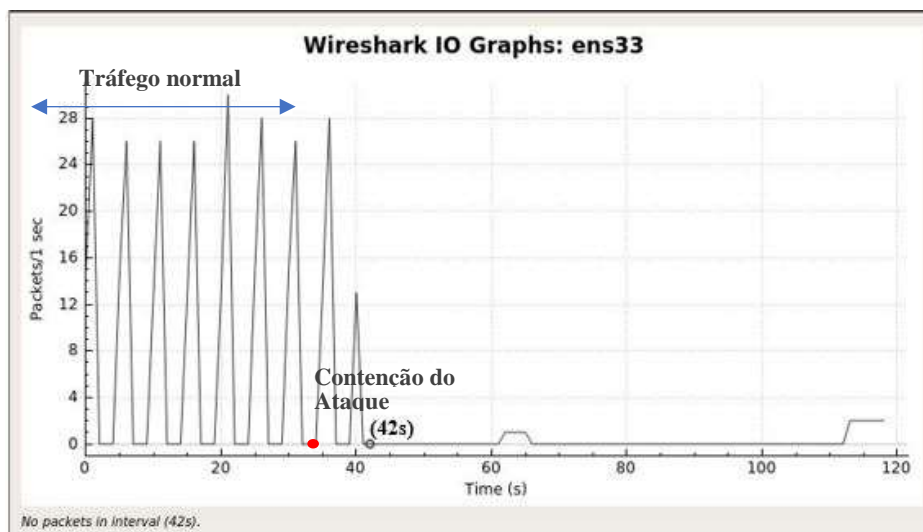
Após a identificação do ataque e geração dos alertas, o *Suricata* solicita ao *Guardian* o bloqueio de tráfegos ICMP dentro da rede local do CAARF-SDN. O Gráfico 3 e 4 apresentam o monitoramento da *interface* do Controlador SDN durante o processo do ataque e contenção.

Gráfico 3 – Análise da execução do ataque ICMP Flood *Smurf* no Controlador ONOS no momento do início da contenção do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

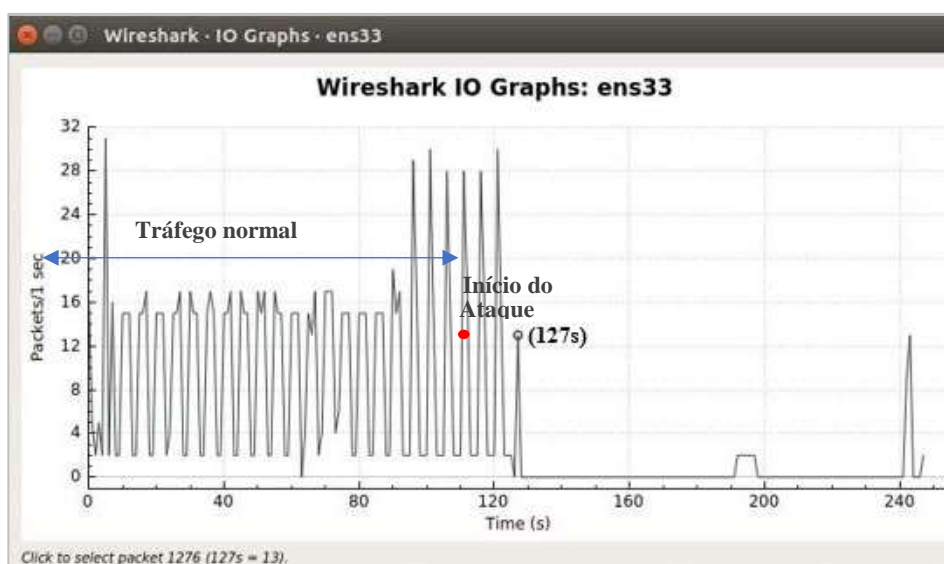
Gráfico 4 – Análise da execução do ataque ICMP Flood *Smurf* no Controlador ONOS no momento exato do bloqueio do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

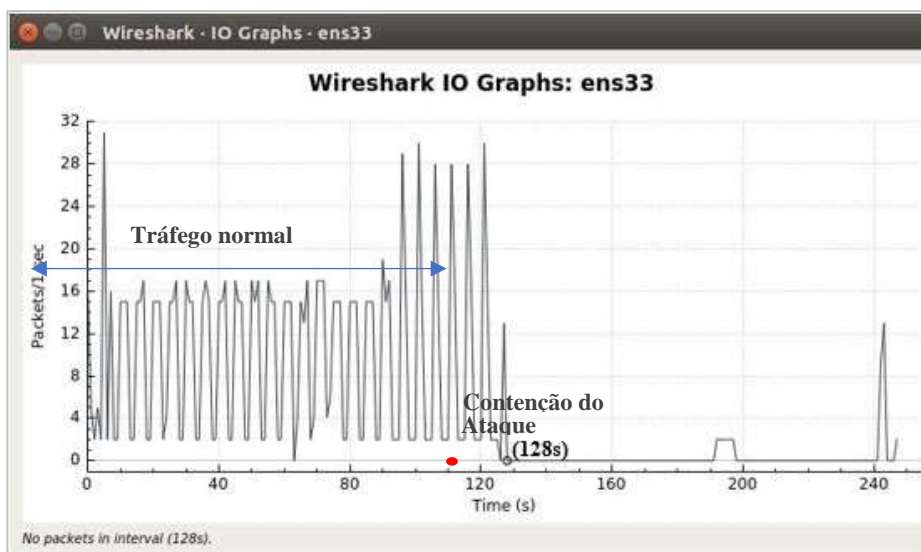
A contenção do ataque na interface do ens33 do Controlador SDN é identificada e ocorre aos 40 segundos e o bloqueio é efetivado aos 42 segundos. O conjunto de mecanismos de segurança implementados levaram apenas 2 segundos para identificar e neutralizar a o ataque de forma eficiente, normalizando o fluxo do tráfego na interface. Os Gráfico 5 e 6 demonstram a contenção do ataque na *interface* ens33 do host figurativo do arcabouço CAARF-SDN.

Gráfico 5 – Análise da execução do ataque ICMP Flood Smurf no hospedeiro de referência do arcabouço CAARF-SDN no momento do início da contenção do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

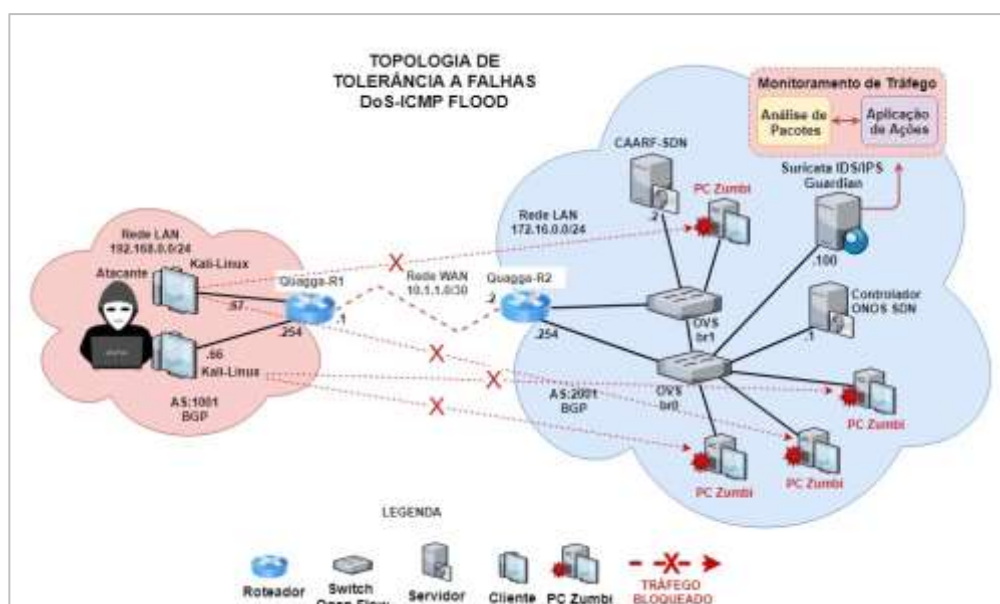
Gráfico 6 – Análise do ataque ICMP Flood *Smurf* no hospedeiro de referência do arcabouço CAARF-SDN no momento exato do bloqueio da contenção do ataque



Fonte: Elaboração própria do autor desta dissertação (2019).

Assim como no resultado do Controlador SDN, a identificação e contenção do ataque na *interface* do host de referência do arcabouço CAARF-SDN ocorreu de forma eficiente e levou apenas 1 segundo para aplicar o bloqueio do ataque, sendo iniciado no segundo 127 e concluindo no segundo 128 da análise, estabilizando o tráfego na interface do arcabouço. Uma visão geral e conceitual desse cenário de contenção de ataque do tipo *Smurf* ICMP Flood é apresentado na Figura 48.

Figura 48 – Topologia Conceitual do cenário de tolerância a falhas com o Suricata

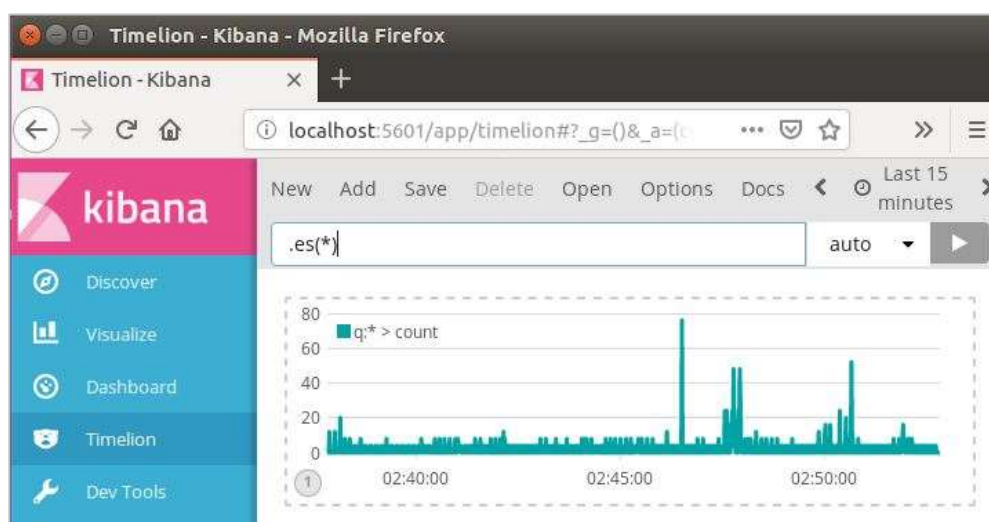


Fonte: Elaboração própria do autor desta dissertação (2019).

Durante todo o experimento, os alertas gerados e armazenados pelo Suricata foram utilizados pelas ferramentas Elasticsearch e Logstash para interagir com o Kibana. Através de arquivos de *logs* json, foi possível integrar uma interface gráfica intuitiva com todos os *logs* detalhados do monitoramento do ambiente de rede CAARF-SDN, através da API do Kibana como. Os alertas são registrados no Kibana, gerando gráficos sobre os *logs* em tempo real, organizando a apresentação dos dados de forma mais elegante e de fácil compreensão.

A Figura 48 apresenta o gráfico gerado em tempo real do monitoramento efetuado pelo Suricata IDS no ambiente de rede do arcabouço CAARF-SDN.

Figura 49 – Interface do Kibana com o gráfico de monitoramento da rede em tempo real baseado nos *logs* do Suricata



Fonte: Elaboração própria do autor desta dissertação (2019).

O gráfico na Figura 49 apresenta o fluxo dos alertas baseados em linha temporal, entretanto é possível configurar filtros para apresentação de gráficos baseados em diversas variáveis através de formulas aplicadas com os parâmetros desejados. A expressão “.es(*)” por exemplo, apresenta um gráfico com todos os parâmetros identificados.

Além dos gráficos em tempo real, o Kibana fez a leitura de vários parâmetros da rede através do arquivo de *log* json da ferramenta Suricata, apresentando um relatório detalhado dos alertas criados pelo monitoramento, como é possível observar na Figura 50.

Figura 50 – Logs de alertas detalhados gerados pelo Kibana baseados na leitura de arquivos json do Suricata IDS



Fonte: Elaboração própria do autor desta dissertação (2019).

Os *logs* aparecem organizados por data e hora de ocorrência, e apresentam diversos detalhes como portas, protocolos, endereçamento IP de origem e destino, assim como tipo do evento. Ao expandir os *logs*, mais detalhes podem ser verificados, possibilitando uma análise forense em cima do comportamento do tráfego da rede.

5.6 CENÁRIO 3 - ALTA DISPONIBILIDADE NA REDE CAARF-SDN

O cenário de alta disponibilidade é proposto nessa dissertação como um complemento aos mecanismos de segurança aplicados nos experimentos do cenário anterior. O objetivo desse cenário é a implementação de mais uma camada de segurança, com o objetivo de garantir que os dispositivos críticos, assim como seus serviços e recursos permaneçam sempre disponíveis na rede.

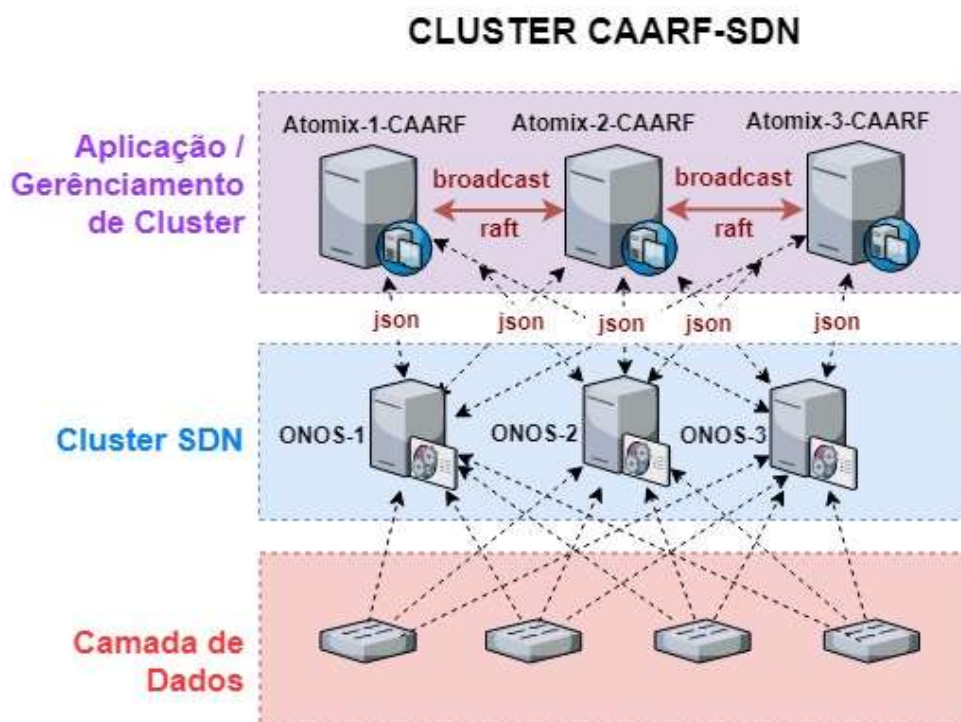
De acordo com a pesquisa em (SAHOO; SAHOO; PANDA, 2016) um das principais características das redes SDN é a centralização do plano de controle que pode ser uma abordagem controversa, dependendo da implementação do Controlador SDN, uma vez que toda a disponibilidade da rede pode ser comprometida em casos onde um único dispositivo seja responsável por todo o plano de controle da rede. Ainda que desde a especificação inicial do

protocolo *OpenFlow* tenha sido discutido a possibilidade de implementação distribuída do Controlador SDN, os mecanismos necessários para essa distribuição só começaram a ser especificados a partir da versão 1.2 do *OpenFlow* (SPALLA et al., 2015). Desde a especificação da versão 1.2 do *OpenFlow* os dispositivos encaminhadores da rede SDN conseguem reconhecer e se conectar de forma simultânea com diversos Controladores, entretanto, nem todos os Controladores SDN atualmente possuem recursos para trabalhar de forma distribuída, sendo necessário um planejamento adequado para garantir a alta disponibilidade em ambiente de redes SDN (SAHOO; SAHOO; PANDA, 2016).

A abordagem de alta disponibilidade em redes definidas por software é de grande relevância no contexto híbrido do CAARF-SDN, tendo em vista que o projeto do arcabouço foi pensado principalmente para estabelecer interações com ambientes arquitetura SDN. Sendo assim, um planejamento distribuído é fundamental para garantir maior disponibilidade do ambiente de rede do arcabouço CAARF-SDN e para fomentar essa abordagem em redes híbridas, onde o ambiente é mais complexo devido a coexistência de diversas tecnologias. Nesse contexto, o arcabouço CAARF-SDN deve fornecer um conjunto de soluções que viabilizam qualidade de serviço e garantias de aspectos de segurança em redes, além da engenharia de tráfego para o qual foi projeto inicialmente.

No projeto atual do CAARF-SDN, a implementação do arcabouço é abordada de forma independente, desacoplada do Controlador SDN (BADARÓ NETO et al., 2018), buscando maior autonomia do arcabouço. Dessa forma, a proposta desse cenário é a criação de uma sub camada dentro da camada de aplicação, com o objetivo de mesclar as funcionalidades do CAARF-SDN com a possibilidade de organização e gerenciamento de *clusters* SDN formados na camada de controle.

A Figura 51 apresenta de forma conceitual, o modelo proposto para promover alta disponibilidade no contexto de rede do arcabouço CAARF-SDN.

Figura 51 – Modelo Conceitual da arquitetura de *cluster* para a rede CAARF-SDN

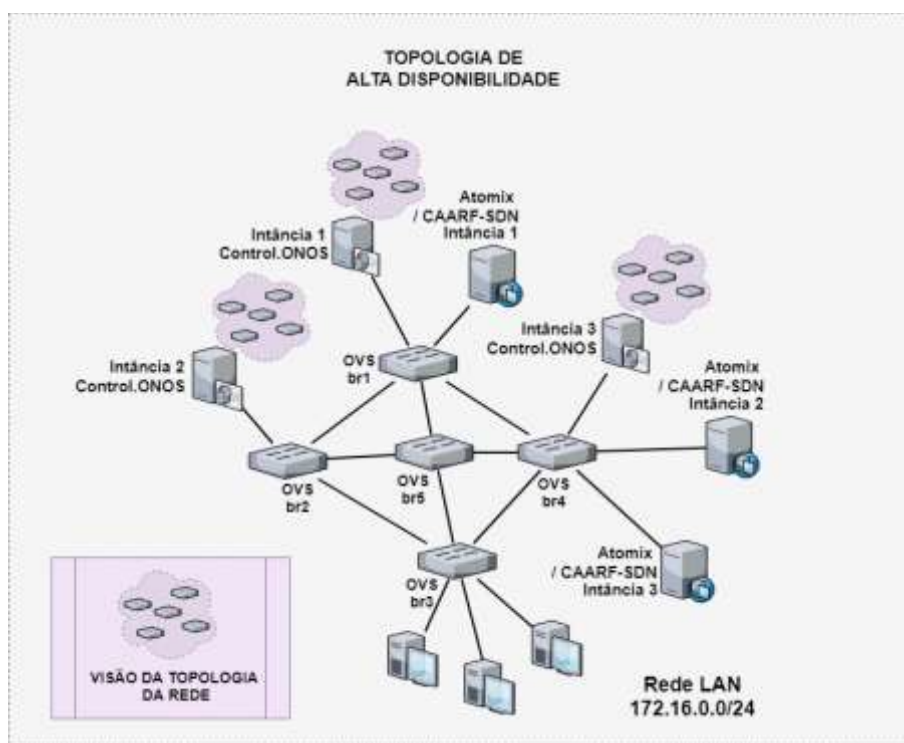
Fonte: Elaboração própria do autor desta dissertação (2019).

Além de possibilitar o gerenciamento distribuído, a camada de aplicação é aprimorada com recursos que viabilizam a formação de cluster na camada de controle. Os gerenciadores por sua vez também formam um cluster na camada de aplicação e trocam informações referentes ao estado dos clusters, suas características e funções. A grande vantagem dessa abordagem de clusters fisicamente desacoplados é a escalabilidade na camada de controle, uma vez que viabilizam a implementação de diversas instâncias do Controlador SDN promovendo alta disponibilidade na camada de controle e na camada de aplicação (SPALLA et al., 2015; HALTERMAN, 2018).

A implementação dos recursos distribuídos é feita através da ferramenta Atomix na camada de aplicação. O Atomix viabiliza a leitura das informações dos diversos controladores SDN através de mensagens baseadas em parâmetros mapeados em arquivos json, que informam quais Controlador SDN fazem parte do *cluster*, aos mesmo tempo que atualizam o estado do *cluster* verificando a integração de novas instâncias ou a falha de algum controlador pertencente ao *cluster* (HALTERMAN, 2018). Todos os controladores têm acesso a existência a informações das instancias vizinhas através do sistema de atualização dinâmica do Atomix, dessa forma é possível saber quando um controlador está disponível ou indisponível no *cluster*. Na camada de dados, os *switches OpenFlow* percebem a camada de controle de forma

transparente, interagindo de forma logicamente centralizada, ainda que os Controladores SDN estejam fisicamente distribuídos. Dessa forma, no caso de ocorrência de falhas em um dos controladores, os demais componentes do cluster continuam assumindo o papel, uma vez que todos tem a mesma visão da topologia da rede (HALTERMAN, 2018). A Figura 52 ilustra um cenário conceitual de aplicação do cluster CAARF-SDN.

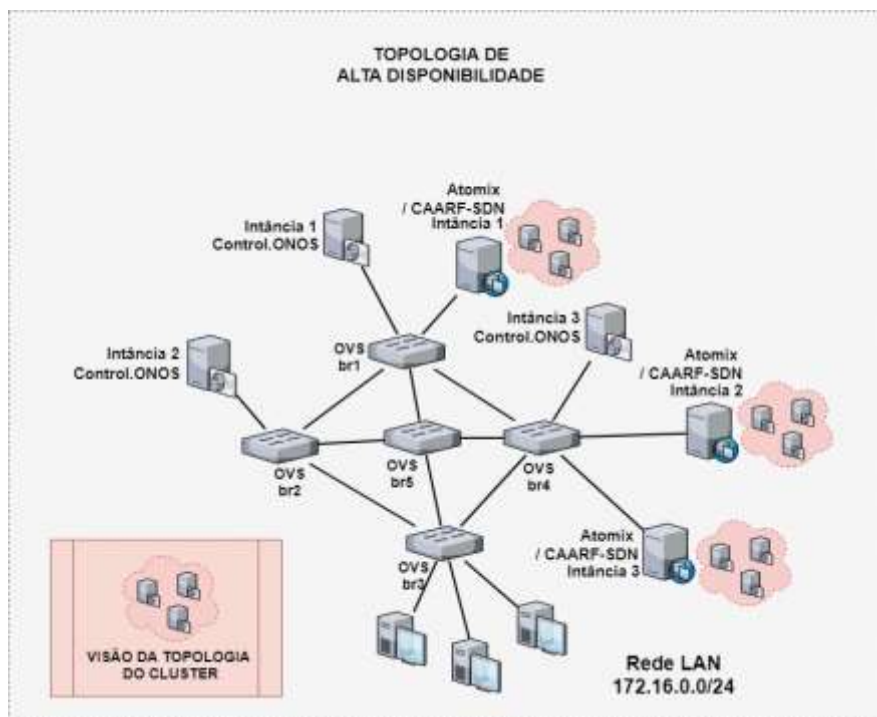
Figura 52 – Topologia conceitual do cenário de alta disponibilidade com a visão compartilhada dos Controladores SDN sobre a topologia da rede



Fonte: Elaboração própria do autor desta dissertação (2019).

Cada instância da camada de controle mantém informações atualizadas sobre a topologia da rede, compartilhando esse estado e atuando de forma simultânea e transparente na programação de fluxo *OpenFlow* nas tabelas dos switches da rede. Na camada de aplicação, as instâncias do arcabouço CAARF-SDN executam os agentes da ferramenta Atomixa para manter e coletar o estado do cluster da camada de controle, possibilitando que os Controladores SDN consigam se comunicar, como é ilustrado na Figura 53.

Figura 53 – Topologia conceitual do cenário de alta disponibilidade com a visão compartilhada dos arcabouços Atomix/CAARF-SDN sobre a topologia da rede



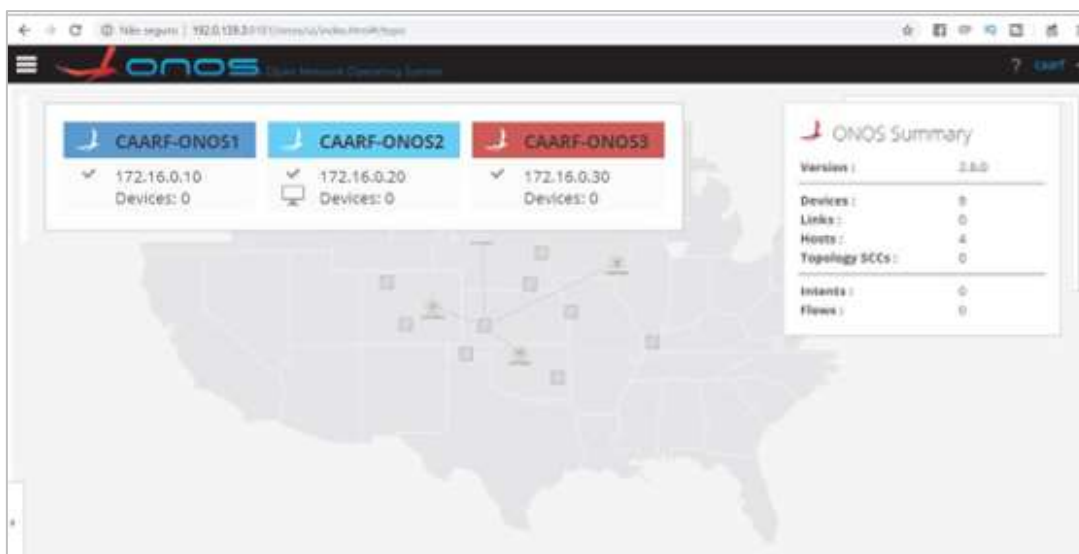
Fonte: Elaboração própria do autor desta dissertação (2019).

Essa arquitetura possibilita que outras ferramentas na camada de aplicação como é o caso do CAARF-SDN, possam interagir com o cluster de controladores de forma transparente.

Nesse trabalho é proposto um cenário simples de cluster, onde é realizada a configuração de um cluster com três instâncias na camada de controles, gerenciada por três instâncias de hospedeiros figurativos do arcabouço CAARF-SDN. Cada instância de controlador executa o sistema operacional de rede ONOS, devido a sua capacidade trabalhar de forma distribuída.

A Figura 54 apresenta o painel do controlador ONOS com a visualização da topologia da rede e das instâncias de controladores ativos para a rede SDN.

Figura 54 - Interface do Controlador ONOS com a visão compartilhada da rede e das instâncias do *cluster* de Controle SDN



Fonte: Elaboração própria do autor desta dissertação (2019).

Na GUI, cada instância é representada por uma cor e identificadas por seus respectivos nomes e endereços IPs vinculados, no entanto, é possível ter acesso a informações mais detalhas das instâncias através da página *Cluster Nodes*, como é apresentado na Figura 55.

Figura 55 – Detalhes do *cluster* de controladores na interface do ONOS

ACTIVE	STARTED	NODE ID	IP ADDRESS	TCP PORT	LAST UPDATED
✓	✓	caarf-onos1	172.16.0.10	9876	10:08:25 AM -07:00
✓	✓	caarf-onos2	172.16.0.20	9876	9:41:15 AM -07:00
✓	✓	caarf-onos3	172.16.0.30	9876	9:41:52 AM -07:00

Fonte: Elaboração própria do autor desta dissertação (2019).

As informações apresentadas destacam o estado de cada controlador, seu ID, IPs, porta TCP e a últimas atualizações identificada por cada instância. É fundamental destacar que a visão da rede e as informações apresentadas são as mesmas para todos as instâncias controladores que fazem parte do *cluster*.

A arquitetura de alta disponibilidade viabiliza ao ambiente CAARF-SDN um complemento eficiente ao ambiente de tolerância a falhas apresentado na sessão 5.5, uma vez

que garante a continuidade dos serviços e recursos da rede mesmo em caso de falhas físicas ou lógicas.

Para avaliação dos mecanismos impostos no presente cenário, foi simulado uma queda de link em uma das instâncias para observar o comportamento, o tempo de resposta e a o estado do serviço SDN na rede. Na Figura 56 pode-se observar continuidade da rede após uma falha intencional no experimento, provocada desabilitando a interface de rede do controlador CAARF-ONOS1.

Figura 56 – Estado do *cluster* de controladores após falha em uma das instâncias



Fonte: Elaboração própria do autor desta dissertação (2019).

Mesmo com a falha em uma das instâncias controladoras, a camada de controle continua operacional, mantendo o mesmo estado da rede, na transparecendo nenhuma anomalia para os dispositivos da camada de dados. Outra característica importante é a capacidade de implementar uma nova instância ou reativar instâncias em falha, sem alterar ou prejudicar o estado atual da rede, uma vez que cada nova instância habilitada recebe atualizações da topologia da rede e seu estado de funcionamento (HALTERMAN, 2018; ATOMIX, 2019; DHIYAULHAQ, 2019).

5.7 DISCUSSÃO E LIÇÕES APRENDIDAS

Os experimentos realizados neste capítulo demonstram a necessidade de abordagens estratégicas relacionadas à segurança no contexto de redes híbridas baseadas na arquitetura SDN, em específico, no ambiente de rede para aplicação do arcabouço CAARF-SDN.

A presente sessão apresenta uma análise comparativa dos resultados obtidos nos experimentos executados nessa dissertação, por meio do sistema CVSS com o objetivo de mensurar o impacto das vulnerabilidades e ameaças identificadas, assim como a eficiência dos mecanismos de contenção implementados.

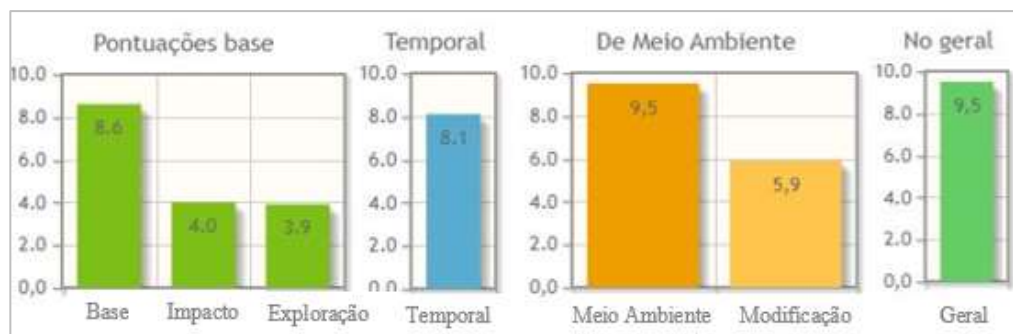
O CVSS possibilita a captura das características mais relevantes das vulnerabilidades fornecendo uma pontuação que reflete na percepção do impacto das ameaças e vulnerabilidades de sistemas computacionais, possibilitando maior expertise em relação a tomada de decisões e redução de riscos (FIRST, 2019).

A estratégia do sistema CVSS se baseia na captação de informações sobre o ambiente que se deseja avaliar e no cálculo de métricas relacionadas à segurança da informação que resultam na pontuação quantitativa que indica o grau dos riscos relacionados a vulnerabilidades existentes em um sistema. As métricas CVSS são divididas em três grupos sendo eles, “pontuação base”, “temporal” e “de meio ambiente”.

- a) **Pontuação base:** Esse grupo de métricas representam as características de vulnerabilidades que são constantes ao longo do tempo e é comporta por dois conjuntos, sendo as métricas de exploração de ativos e as métricas relacionadas ao impacto da ameaça. O conjunto de métrica de exploração refletem as técnicas pelas quais a vulnerabilidade pode ser explorada, enquanto o conjunto de métricas de impacto refletem as consequências de uma exploração bem sucedida em um ativo;
- b) **Temporal:** As métricas temporais refletem as características de vulnerabilidades que podem mudar de acordo com o tempo. Busca avaliar o nível de exploração em relação ao estado atual das técnicas de exploração, ou seja, a maturidade dos códigos maliciosos e o nível de confiança das soluções existentes para tais vulnerabilidades, e;
- c) **Métricas ambientais:** Esse último grupo de métricas refletem o impacto do ambiente onde o ativo está inserido em sua vulnerabilidade, ou seja, avalia se o ambiente auxilia ou não na exploração do ativo.

No Gráfico 7 é possível verificar os resultados da avaliação CVSS para os cenários onde não foram implementados mecanismos de proteção.

Gráfico 7 – Resultado da análise de riscos CVSS no cenário de rede do CAARF-SDN antes da implementação dos mecanismos de segurança

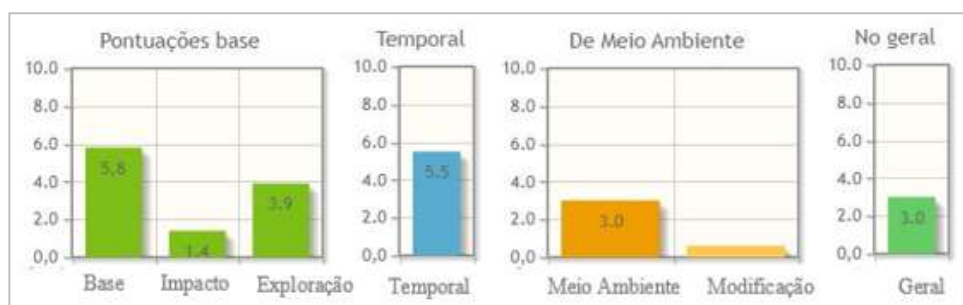


Fonte: Elaboração própria do autor desta dissertação (2019).

Na pontuação de base é apresentada o nível de facilidade de exploração das vulnerabilidades dos cenários propostos nessa dissertação, assim como o nível de impacto dessas vulnerabilidades ao serem exploradas. A pontuação temporal apresenta o alto risco de mudança e inovação nos métodos de ataque utilizados pelas ameaças apresentadas nessa dissertação, ou seja, a capacidade que essas ameaças tem de se adaptar ao ambiente vulnerável. Por fim as pontuações das métricas de meio ambiente demonstram como o ambiente de rede dos cenários apresentados podem influenciar nos ataques ao Controlador SDN e ao CAARF-SDN. Os resultados apresentados no Gráfico 7 representam as análises realizadas por foi o sistema.

A pontuação geral CVSS para os experimentos realizados nessa dissertação, deixa evidente os altos índices de riscos existentes no contexto do ambiente padrão de rede híbrida do CAARF-SDN, apresentados em cada uma das métricas avaliadas do CVSS, comprovados com o sucesso resultante dos diversos ataques realizados na rede do arcabouço. Da mesma forma, é possível verificar o impacto da implementação dos mecanismos de segurança nos experimentos realizados, através da avaliação de riscos CVSS apresentada no Gráfico 8.

Gráfico 8 – Resultado da análise de riscos CVSS nos cenários de rede do CAARF-SDN após a implementação dos mecanismos de segurança



Fonte: Elaboração própria do autor desta dissertação (2019).

Nas comparações dos resultados dos gráficos pode-se observar a queda circunstancial na pontuação dos riscos relacionados aos cenários propostos, evidenciando a eficiência dos mecanismos apresentados e implementados.

Dessa forma, os mecanismos implementados mostraram-se eficientes na mitigação das ameaças impostas resultando na contenção dos ataques realizados. Aspectos de alta disponibilidade e integridade foram validados comprovando a necessidade de uma abordagem distribuída e de análise de tráfego no contexto de rede do arcabouço CAARF-SDN, com o objetivo de torná-lo uma ferramenta viável para garantir aspectos de segurança em ambientes de rede SDN, além de qualidade de serviço e engenharia de tráfego.

Foi apresentado ainda, uma abordagem distribuída que adequa a utilização do arcabouço CAARF-SDN atribuindo novas funcionalidades e uma nova perspectiva para projetos futuros do projeto. Assim como foi modelado e proposto um cenário de rede SDN viável para implementações, testes e experimentos futuros de projetos relacionados ao CAARF-SDN.

5.8 CONCLUSÃO

Nesse capítulo foram apresentados diversos cenários e experimentos que demonstraram as vulnerabilidades e riscos existentes no contexto de rede do arcabouço CAARF-SDN, assim como foi implementado um conjunto de mecanismos de segurança propostos com o objetivo de garantir aspectos de segurança na rede do CAARF-SDN e promover garantias de serviços e recursos fornecidos nesse ambiente de rede.

Os experimentos realizados envolveram a exploração de vulnerabilidades dos componentes críticos da rede CAARF-SDN, através de ataques que se mostraram efetivos nos cenários padrões (*default*), onde os níveis de segurança eram mínimos, evidenciando a necessidade de um planejamento adequado de segurança no contexto do CAARF-SDN.

6 CONCLUSÕES E PERSPECTIVAS FUTURAS

Nesse último capítulo são apresentadas as principais conclusões relacionadas a esse trabalho de mestrado, assim como algumas perspectivas futuras, propostas com o objetivo de direcionar a continuação e aprimoramento dessa pesquisa.

6.1 CONCLUSÕES

A presente dissertação de mestrado propôs a discussão sobre os riscos e vulnerabilidades existentes no contexto dos novos paradigmas de redes de computadores. Para isso, foram apresentadas as principais ameaças e aspectos vulneráveis existentes nas redes baseadas no paradigma SDN, assim como os principais conceitos relacionados à segurança da informação, aplicados aos ambientes de redes de computadores.

Foram definidas e catalogadas as principais vulnerabilidades e ameaças identificadas no contexto do CAARF-SDN, com o objetivo de compreender e mensurar os riscos aos quais o ambiente de rede do arcabouço está exposto. Dessa forma, foi possível viabilizar o planejamento para implementação de mecanismos para garantir aspectos de segurança tais como confidencialidade, integridade, disponibilidade, não-repúdio e autenticidade. Tais aspectos influenciam diretamente na Qualidade de Serviço (QoS), Qualidade de Dispositivos (QoD) e na Qualidade de Experiência (QoE), monitorados e manipulados pelo arcabouço CAARF-SDN.

Com o objetivo de validar e viabilizar a implementação desses mecanismos de segurança, foram propostos alguns cenários conceituais que serviram como referência para o desenvolvimento dos experimentos realizados nesse trabalho. Em um segundo momento, esses cenários foram implementados através de um ambiente de rede SDN virtualizado, com o objetivo de representar a rede do arcabouço CAARF-SDN.

O primeiro experimento desenvolvido propôs a realização de ataques do tipo DDoS (SYN Flood, ICMP Flood e *Smurf*) no ambiente de rede virtualizado e configurado de forma padrão (*default*), sem nenhum tipo de planejamento ou implementação de mecanismos de segurança específicos para tratar as vulnerabilidades e mitigar as ameaças.

Nesse primeiro experimento, todos os ataques realizados se mostraram efetivos e tiveram um grande impacto em dispositivos críticos da rede, como o controlador SDN e o hospedeiro representativo do arcabouço CAARF-SDN, afetando o comportamento e a disponibilidade de serviços importantes para o funcionamento adequado da rede.

O segundo experimento denominado de tolerância a falhas, teve como principal objetivo, a validação de mecanismos de segurança, implementados com o propósito de garantir aspectos como integridade, disponibilidade, não-repúdio e autenticidade. Para esse propósito, foram utilizadas ferramentas como Suricata IDS e Guardian, no intuito de monitorar e auditar todo o tráfego de entrada e saída da rede virtualizada do CAARF-SDN, o que possibilitou a reação automatizada no processo de contenção de ataques. Para uma leitura e análise gráfica em tempo real dos *logs* de monitoramento, foi utilizado o conjunto de ferramentas denominado Elasticsearch.

No segundo experimento, o ambiente virtualizado para representar a rede do arcabouço CAARF-SDN foi submetida aos mesmos tipos de ataques realizados no primeiro experimento. No entanto, os mecanismos de segurança implementados se mostraram eficientes no processo de monitoramento, detecção e contenção dos ataques, proporcionando uma camada de proteção em tempo real para a rede, garantido o seu funcionamento adequado, assim como serviços e dispositivos críticos.

O terceiro experimento propôs recursos para viabilizar a alta disponibilidade dos recursos e serviços críticos no ambiente de rede do arcabouço CAARF-SDN. Inicialmente foi proposto um modelo conceitual da arquitetura base para esse experimento. Posteriormente, um segundo ambiente virtualizado de rede SDN foi implementado de acordo com as especificações do modelo conceitual apresentado.

Na implementação do esquema alta disponibilidade no contexto de rede do arcabouço CAARF-SDN, foi utilizado o controlador ONOS devido a sua compatibilidade com recursos para integração de *clusters* na camada de controle da rede. O sistema de *clusters* dos controladores foi implementado através da ferramenta Atomix, responsável por gerenciar cada uma das instâncias dos controladores.

O esquema proposto e implementado nesse terceiro experimento, viabilizou a criação de clusters tanto na camada de controle como na camada de aplicativo, possibilitando a alta disponibilidade de todos os recursos e dispositivos críticos no ambiente de rede do CAARF-SDN. Para testar a efetividade do esquema implementado, foram criadas falhas intencionais em alguns dispositivos críticos (controlador SDN e hospedeiro representativo do CAARF-SDN), com o objetivo de paralisar seus serviços e deixá-los indisponíveis. Com isso, pode-se comprovar a eficiência do esquema de alta disponibilidade, uma vez que mesmo após a retirada de determinada instância seja na camada de controle ou seja na camada de aplicativo e

gerenciamento do cluster, as demais instâncias assumiram o papel das instâncias em estado de falha, possibilitando a continuidade dos serviços da rede.

Além dos resultados coletados e gráficos comparativos gerados através da ferramenta Wireshark, o sistema CVSS foi utilizado como um segundo método de validação, para medir a efetividade dos mecanismos de segurança implementados durante os experimentos realizados nessa dissertação, através da avaliação de métricas relacionadas aos riscos e vulnerabilidades identificados e catalogados no ambiente de rede do CAARS-SDN.

Através dos resultados apresentados nos gráficos comparativos do CVSS, é possível verificar a efetividade dos mecanismos implementados no ambiente de rede do CAARF-SDN, uma vez que apresenta a diminuição dos riscos relacionados aos ataques e vulnerabilidades avaliadas.

Algumas vantagens são consideradas nas soluções propostas nesse trabalho como processos automatizados de monitoramento do tráfego, detecção e prevenção de ameaças em tempo real, uma vez que a identificação e reação aos ataques ocorrem em um tempo relativamente baixo, possibilita a mitigação das ameaças de forma eficiente e dinâmica. A independência dos mecanismos propostos também se apresenta como efetiva vantagem, uma vez que podem ser adaptada para atender a diferentes ambientes de redes, sejam elas legadas ou baseadas nos novos paradigmas.

Apesar dos benefícios apresentados, é importante salientar que os mecanismos modelados e implementados não abrangem todas as abordagens de ameaças existentes no contexto do arcabouço CAARF-SDN, uma vez que todos os esforços foram direcionados para vulnerabilidades e ameaças consideradas críticas, que representam alto risco para o funcionamento da rede CAARF-SDN e seus serviços.

Por fim, a relevância da presente dissertação, pode ser evidenciada pela discussão orientada à análise de vulnerabilidades e ameaças, assim como pela modelagem, implementação e validação de mecanismos de segurança no contexto de rede do CAARF-SDN, uma vez que todas as outras pesquisas relacionadas ao CAARF-SDN não abrangem esses aspectos. Dessa forma foi possível contribuir com o projeto CAARF-SDN, garantindo a implementação segura do arcabouço, seus recursos e serviços.

6.2 PERSPECTIVAS FUTURAS

Nessa seção são apresentadas algumas das perspectivas futuras identificadas com o objetivo de direcionar a continuidade dessa pesquisa, sendo as principais perspectivas identificadas:

- a) Implementação da interação e troca de informações de contexto entre as instâncias do CAARF-SDN na camada de aplicativo dentro do *cluster*;
- b) Implementação de monitoramento inteligente, detecção e prevenção de ataques através da aprendizagem de máquina com ferramentas como Wazuh e Elastic Stack;
- c) Modelagem, implementação e validação de mecanismos de segurança para mitigação de ataques do tipo *Man In The Middle* dentro do contexto de rede do arcabouço CAARF-SDN;
- d) Modelagem, implementação e validação de mecanismos de segurança para tratar e mitigar ataques baseados em *malwares* no contexto de rede do arcabouço CAARF-SDN;
- e) Modelagem, implementação e validação de mecanismos de segurança para tratar e mitigar ataques baseados em falsificação de dispositivos válidos no contexto de rede do arcabouço CAARF-SDN;
- f) Integração de mecanismos de análise de informações contextuais em grandes volumes de dados baseados em *Big Data* e *Data Warehouse*, focado em identificação de ameaças e análise do comportamento do tráfego no contexto de rede do CAARF-SDN.

REFERÊNCIAS

- ABDELSALAM, A. M.; EL-SISI, A. B.; K, V. R. Mitigating ARP Spoofing Attacks in Software-Defined Networks. **ICCTA 2015, Alexandria, Egypt**, n. oct. 2015.
- ABNT, N. I. I. 27002. **Tecnologia da Informação-Técnicas de Segurança – Código de Prática para controles de segurança da informação ISO/IEC 27002**, 2013b. Disponível em: <https://kupdf.net/downloadFile/58f925dfdc0d60820cda9805> Acesso em: 19 jun. 2019.
- ATOMIX. **Cluster Configuration**. Disponível em: <https://atomix.io/docs/latest/user-manual/cluster-management/cluster-configuration/>. Acesso em: 19 jun. 2019.
- BADARÓ NETO, F. J. B. V. et al. Context-Based Dynamic Optimization of Software Defined Networks. *In: INTERNATIONAL WORKSHOP ON ADVANCES IN ICT INFRASTRUCTURE AND SERVICE*, 2018. **Proceedings [...]** 2018. p. 142–149.
- BADOTRA, S.; SINGH, J. OpenDaylight as a controller for Software Defined Networking. **International Journal of Advanced Research in Computer Science**, v. 8, n. 5, p. 7, 2017.
- BALAGOPAL, D.; RANI, X. A. K. A Technique for a Software-Defined and Network-based ARP Spoof Detection and Mitigation. **International Journal of Applied Engineering Research**, v. 13, n. 20, p. 14823–14826, 2018.
- BASSO, T. **Uma abordagem para avaliação da eficácia de scanners de vulnerabilidades em aplicações web**. 2010. Dissertação (Mestrado de Engenharia Elétrica)- Universidade Estadual de Campinas - UNICAMP, Campinas, São Paulo, 2010.
- BENTON, K.; CAMP, L. J.; SMALL, C. OpenFlow vulnerability assessment. *In: ACM SIGCOMM WORKSHOP ON HOT TOPICS IN SOFTWARE DEFINED NETWORKING - HOTSDN '13*, 2., 2013. **Proceedings [...]** 2013. p. 151.
- BERALDO, L. et al. Análise Conceitual e Comparativa entre Sistemas de Detecção de Intrusão. *In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS - SBRC*, 36., 2018. **Proceedings [...]** 2018.
- BOGAERTS, P.; XXRADAR, A. **HPING tutorialradarhack**, 2003. Disponível em: http://www.radarhack.com/dir/papers/hping2_v1.5.pdf Acesso em: 16 jul. 2019
- BON, C. H. B. **Análise de Parametrizações de Qualidade de Serviço em Uma Rede Metropolitana para Tráfego VoIP**. 2008. Dissertação (Mestrado em Engenharia Elétrica)- Universidade de Brasília, Brasília, Distrito Federal, 2008.
- BRUNNSTRÖM, K. et al. Qualinet White Paper on Definitions of Quality of Experience. **Qualinet White Paper on Definitions of Quality of Experience Output from the fifth Qualinet meeting**, p. 11–13, 2014.
- CAMELO, A.; BEZERRA, D.; SILVA, D. S.; SANTANA, F. L. C.; SANTOS, G. J. R.; SANTANA, I. A. B. T.; COSTA, M. A. P. Análise Comparativa dos Módulos de IPS / IDS do Suricata e Snort. **Revista Científica Tecnologus**, v. 11, 2017.

CERTSI. **Design and Configuration of IPS , IDS and SIEM in Industrial Control Systems**, 2017. Disponível em: https://www.certs.es/sites/default/files/contenidos/guias/doc/certs_design_configuration_ips_ids_siem_in_ics.pdf Acesso em: 16 jul. 2019

CHAOTIC. **Guardian Active Response for Snort**. Disponível em: <https://www.chaotic.org/guardian/>. Acesso em: 18 jul. 2019.

CHHAJED, S. **Learning ELK Stack**. Birmingham, Mubai: Packt Publishing, 2015.

CLARO, J. **Sistemas IDS e IPS: estudo e aplicação de ferramenta open source em ambiente linux**. Passo Fundo: Instituto Federal de Educação, Ciência e Tecnologia Sul-Rio- Grandense - Ifsul, 2015.

COMITÊ GESTOR DA INTERNET NO BRASIL. **Cartilha de Segurança para Internet, versão 4.0**. [S.l]: [s.n.], 2012.

CORRÊA, J. H. G. et al. SHADE: Uma estratégia seletiva para mitigar ataques DDoS na camada de aplica ao em redes definidas por software. *In: SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES - SBRT*, 34., 2016. **Anais [...]** 2016. p. 964–968.

COSTA, S.; PEREIRA, M. A.; ARA, M. Segurança de Redes de Computadores na Internet Security of Computer Networks on Internet. **Revista Inova Ação**, p. 77–88, 2012.

COUTINHO, M. M. et al. *Revista Gestão em Foco - Edição nº 9 – Ano: 2017*. p. 489–500, 2017.

CVSS. **Sistema de pontuação comum da vulnerabilidade v3.1: Documento de especificação**. Disponível em: <https://www.first.org/cvss/specification-document>. Acesso em: 1 set. 2019.

DEY, A. K.; ABOARD, G. D.; SALBER, D. A Conceptual Framework and a Toolkit for Supporting The Rapid Prototyping of Context-aware Applications. **Human-Computer Interaction**, v. 16, n. 2–4, p. 97–166, 2001.

DHIYALHAQ, Z. **Creating ONOS Cluster 2.0.0**. Disponível em: <https://blog.zufardhiyaulhaq.com/creating-onos-cluster-2-0-0/>. Acesso em: 30 jun. 2019.

DUNGAY, D. **Software Defined Networking (SDN) Explained**. Disponível em: <https://www.commsbusiness.co.uk/features/software-defined-networking-sdn-explained/>. Acesso em: 27 maio. 2019.

ERICKSON, D. The Beacon OpenFlow Controller. **ACM SIGCOMM**, p. 6, 2013.

FERREIRA, C. C. **Análise Comparativa de Controladores para Redes Definidas por Software de Classe Carrier Grade**. Dissertação de Mestrado - Curso de Ciência da Computação, Universidade Federal de Uberlândia - UFU, Uberlândia, Minas Gerais, 2016.

FIRST. **Common Vulnerability Scoring System version 3.1: Specification Document**. Disponível em: <https://www.first.org/cvss/v3.0/specification-document>. Acesso em: 27 ago.

2019.

GAIGOLE, M. S.; KAMALTAI, S.; KALYANKAR, M. A. The Study of Network Security with Its Penetrating Attacks and Possible Security Mechanisms. **International Journal of Computer Science and Mobile Computing**, v. 45, n. 5, p. 728–735, 2015.

GIL, A. C. **Métodos e Técnicas de Pesquisa Social**. Atlas S.A. ed. São Paulo: [s.n.], v. 6, 2008.

GNS3. **Getting Started with GNS3**. Disponível em: https://docs.gns3.com/1PvtRW5eAb8RJZ11maEYD9_aLY8kkdhgaMB0wPCz8a38/index.html. Acesso em: 15 jul. 2019.

GOMES, V.; ISHIMORI, A.; FARIAS, F. Isolamento de Recursos de Rede em Ambientes virtuais Baseados em Openflow/SDN. **Sbrc2013.Unb.Br**, p. 91–104, 2013.

GOULART, P. et al. NetFPGA: Processamento de Pacotes em Hardware. **NetFPGA**, p. 1–48, 2015.

GUEDES, D. et al. Redes Definidas por Software: Uma Abordagem Sistêmica para o Desenvolvimento de Pesquisas em Redes de Computadores. in: **Minicursos do XXX Simpósio Brasileiro de Redes de Computadores - SBRC**, p. 160–210, 2012.

HALTERMAN, J. **Cluster Configuration in Owl**. Disponível em: <https://wiki.onosproject.org/pages/viewpage.action?pageId=28836788>. Acesso em: 30 jun. 2019.

HPING. **Home**. Disponível em: <http://www.hping.org/>. Acesso em: 19 jul. 2019.

JAKMA, P.; LAMPARTER, D. Introduction to the Quagga Routing Suite. **IEEE Network**, v. 28, n. 2, p. 42–48, 2014.

KAUR, K.; SINGH, J.; GHUMMAN, N. S. Mininet as Software Defined Networking Testing Platform. **International Conference on Communication, Computing & Systems (ICCCS–2014)**, n. December, p. 3–6, 2014.

KHAN, S. A. A STRIDE Model based Threat Modelling using Unified and-Or Fuzzy Operator for Computer Network Securit. **International Journal of Computing and Network Technology**, v. 5, n. 1, p. 13–20, 2017.

KHURSHID, A. et al. Veriflow. **ACM SIGCOMM Computer Communication Review**, v. 42, n. 4, p. 467, 2012.

KIM, H.; FEAMSTER, N. Improving Network Management With Software Defined Networking. **IEEE Communications Magazine**, v. 51, n. 2, p. 114–119, 2013.

KLOTI, R.; KOTRONIS, V.; SMITH, P. OpenFlow : A Security Analysis. **2013 21st IEEE International Conference on Network Protocols (ICNP)**, p. 6, 2013.

KREUTZ, D. et al. Software-Defined Networking: A Comprehensive Survey. in: **Proceedings of The IEEE (P IEEE)** p. 1–61, 2014.

LAGHARI, K.; CONNELLY, K. Quality Assessment of Multidimensional Video Scalability. **IEEE Communications Magazine**, v. 50, n. 4, p. 38–46, 2012.

LAGHARI, K. U. R.; CRESPI, N.; MOLINA, B.; PALAU, C. E. QoE Aware Service Delivery in Distributed Environment. **Proceedings - 25th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2011**, n. March, p. 837–842, 2011.

LI, D.; HONG, X.; BOWMAN, J. Evaluation of Security Vulnerabilities by Using ProtoGENI as a Launchpad. **GLOBECOM - IEEE Global Telecommunications Conference**, 2011.

LOPEZ, M. E. A. **Uma Arquitetura de Detecção e Prevenção de Intrusão para Redes Definidas por Software**. Dissertação de Mestrado - Curso de Engenharia Elétrica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, p. 72 p., 2014.

MCKEOWN, N. et al. OpenFlow: Enabling Innovation in Campus Networks. **ACM SIGCOMM Computer Communication Review**, v. 38, n. 2, p. 69, 2008.

MENEZES, D. et al. AuthFlow: Um Mecanismo de Autenticacao e Controle de Acesso para Redes Definidas por Software. **Anais do 33º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos- SBRC**, p. 661–674, 2014.

MIGAULT, D.; POURZANDI, M. Identifying and addressing the vulnerabilities and security issues of SDN. **Ericsson Technology Review**, V. 92, n. 7, August, 2015.

MING, D. et al. Application of Movement-related Cortical Potentials in The Field of Motor Rehabilitation. **Nami Jishu yu Jingmi Gongcheng/Nanotechnology and Precision Engineering**, v. 13, n. 6, p. 425–433, 2015.

MOLINA, D.; SILVEIRA, S. R.; SANTOS, F. B. Implantação de Um Ambiente de Segurança de Redes de Computadores : Um Estudo de Caso na Prefeitura Municipal de Palmeira das Missões - RS. 2015.

MORAIS, G. **Análise e Implementação de Sistemas IDS e IPS**. Dissertação de Mestrado - Curso de Engenharia Informática, Universidade de Lisboa, p. 71, 2011.

MOREIRAS, J. A. **Redes Definidas por Software: Do Estado da Arte Tecnológico à Identificação de um Conjunto de Boas Práticas**. Dissertação de Mestrado - Curso de Sistemas de Informações Organizacionais, Setúbal, Portugal, 2016.

MORIMOTO, C. E. **Redes: Guia Prático**. S. Editores, Porto Alegre: [s.n.], 2011.

MUAKAD, C.F.J. **Context-based Dynamic and Adaptive Forwarding Management**. 2015. Dissertação (Mestrado)- UNIFACS Universidade Salvador, Salvador, 2015.

NAKAMURA, E. T. **Segurança da Informação e de Redes**. Londrina - PR: Educacional S.A, 2016.

NAZARIO, D. C.; DANTAS, M. A. R.; TODESCO, J. L. Taxonomia das Publicações sobre Qualidade de Contexto. **Sustainable Business International Journal**, n. 20, 2012.

OLIVEIRA, A. L. C. et al. Context-Aware Adaptive Routing Framework: A Queueing Management Approach. **4rd International Workshop on ADVANCES in ICP Infrastructures and Services**, 2015.

OLIVEIRA, A. L. C. DE. Mecanismo de Notificação Baseado em Contexto para Encaminhamento Adaptativo. **UNIFACS - UNIVERSIDADE SALVADOR**, 2015b.

ONF. SDN Security Considerations in the Data Center. **ONF Solution Brief**, p. 1–12, 2013.

ONF. SDN Architecture for Transport Networks. **ONF White Paper**, p. 17, 2016.

ONOS. **Introdução: O que é o ONOS?** Disponível em: <https://wiki.onosproject.org/display/ONOS/ONOS>. Acesso em: 8 maio. 2019.

PINHEIRO, J. M. DOS S. Ameaças e Ataques aos Sistemas de Informação: Prevenir e Antecipar. **UniFOA**, p. 11, 2007.

PORRAS, P. et al. A Security Enforcement Kernel for OpenFlow Networks. **Proceedings of the first workshop on Hot topics in software defined networks - HotSDN '12**, p. 121, 2012.

PORRAS, P. et al. Securing the Software-Defined Network Control Layer. **Ndss '15**, n. February, p. 8–11, 2015.

RIBEIRO, I. C. G. et al. Segurança em Redes Centradas em Conteúdo: Vulnerabilidades, Ataques e Contramedidas. **Minicurso do Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)**, pág. 101–150, p. 101–150, 2012.

RODRIGUES, J. DE M.; SILVA, R. R. T. DA; LEITE, J. N. DE F. Virtualização e Seus Benefícios para Empresas com Hyper-v; um Estudo de Caso na Indústria. **VIII SEGeT – Simpósio de Excelência em Gestão e Tecnologia**, p. 11, 2011.

RÖPKE, C.; HOLZ, T. SDN rootkits: Subverting network operating systems of software-defined networks. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 9404, p. 339–356, 2015.

ROTHENBERG, CHRISTIAN E. et al. OpenFlow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes. **Cad. CPqD Tecnologia**, p. 65–75, 2012.

SAHOO, K. S.; SAHOO, B.; PANDA, A. A secured SDN framework for IoT. **Proceedings - 2015 International Conference on Man and Machine Interfacing, MAMI 2015**, n. January, 2016.

SCOTT-HAYWARD, S. Design and Deployment of Secure, Robust, and Resilient SDN Controllers. **Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)**, p. July 1–5, 2015.

SCOTT-HAYWARD, S.; NATARAJAN, S.; SEZER, S. A Survey of Security in Software Defined Networks. **IEEE Communications Surveys and Tutorials**, v. 18, n. 1, p. 623–654,

2016.

SCOTT-HAYWARD, S.; O'CALLAGHAN, G.; SEZER, S. SDN Security: A survey. **SDN4FNS 2013 - 2013 Workshop on Software Defined Networks for Future Networks and Services**, p. 1–7, 2013.

SHAIKH, J.; FIEDLER, M.; COLLANGE, D. Quality of experience from user and network perspectives. **Annales des Telecommunications/Annals of Telecommunications**, v. 65, n. 1–2, p. 47–57, 2010.

SHIN, S. et al. Rosemary: A Robust, Secure, and High-Performance Network Operating System. in: **Paper presented at the ACM SIGSAC Conference on Computer and Communications Security (CCS'14)**, Arizona, USA, 2014.

SHU, Z. et al. Security in Software-Defined Networking: Threats and Countermeasures. **Mobile Networks and Applications**, v. 21, n. 5, p. 764–776, 2016.

SILVA, E. L.; MENEZES, E. M. Metodologia da Pesquisa e Elaboração de Dissertação. **Universidade Federal de Santa Catarina**, v. 4 Edição, p. 121, 2005.

SPALLA, E. S. et al. Estratégias para Resiliência em SDN : Uma Abordagem Centrada em Multi-Controladores Ativamente Replicados. **XXXIII Simpósio Brasileiro de Redes de Computadores - SBRC'15**, 2015.

SPINOLA, S. S. Context Management Applied to Adaptive Forwarding Within Convergent Solutions. Dissertação de Mestrado Apresentada à **UNIFACS Universidade Salvador**, 2015.

SURICATA. **What is Suricata**. Disponível em: <https://suricata.readthedocs.io/en/latest/what-is-suricata.html>. Acesso em: 15 jul. 2019.

TANENBAUM, A. S. Redes de petri híbridas diferenciais: Aplicação na modelagem e no gerenciamento dinâmico de energia de redes de sensores sem fio. **Controle y Automacao**, v. 18, n. 3, p. 278–291, 2007.

UNDERDAHL, B.; KINGHORN, G. **Software Defined Networking for Dummies - Cisco Special Edition**. United States: John Wiley & Sons , Inc, 2015.

VACCA, J. R. **Network and System Security**. [s.l: s.n.]. v. 53, 2010.

VIEIRA, V.; TEDESCO, P.; SALGADO, C. Modelos e Processos para o Desenvolvimento de Sistemas Sensíveis ao Contexto. **proceedings of JAI - XXVIII Jornadas de Atualização em Informática**, v. 20, n. 3, p. 381–431, 2009.

WIRESHARK. **Wireshark User's Guide**. Disponível em: https://www.wireshark.org/docs/wsug_html_chunked/. Acesso em: 20 jul. 2019.

WU, J. et al. Context-aware networking and communications: Part 1 [Guest editorial]. **IEEE Communications Magazine**, v. 52, n. 6, p. 14–15, 2014.

WU, W. et al. Quality of experience in distributed interactive multimedia environments:

Toward a theoretical framework. **MM'09 - Proceedings of the 2009 ACM Multimedia Conference, with Co-located Workshops and Symposiums**, p. 481–490, 2009.

XIA, W. et al. A Survey on Software-Defined Networking. **IEEE Communications Surveys & Tutorials**, v. 17, n. 1, p. 72–78, 2014.

YOON, C. et al. Flow Wars: Systemizing the Attack Surface and Defenses in Software-Defined Networks. **IEEE/ACM Transactions on Networking**, v. 25, n. 6, p. 3514–3530, 2017.

ZERKANE, S. et al. Vulnerability Analysis of Software Defined Networking. in: **Foundations and Practice of Security: 9th International Symposium**, p. 97–116, 2017.

ZIMMERMANN, A.; LORENZ, A.; OPPERMANN, R. An operational definition of context. **6th international and interdisciplinary conference on Modeling and using context (CONTEXT'07)**, Fraunhofer Institute for Applied Information Technology, v. 4635 LNAI, p. 558–571, 2007.

PUBLICAÇÕES DO AUTOR

SILVA, R. G., SAMPAIO, P. N. M., (2019). Security Issues In Hybrid Approaches: Context-Based And Software-Defined Networks. In: 7th International Workshop on ADVANCEs in ICT Infrastructures and Services, Praia, Cape Verde, January, 2019.

SILVA, R. G., SAMPAIO, P. N. M., (2020). Mitigating Man in the Middle attacks within Context-based SDNs. In: In: 8th International Workshop on ADVANCEs in ICT Infrastructures and Services, Cancún, México, January, 2020.

APÊNDICE A – CONFIGURAÇÃO REGRAS DO SURICATA**Caarf.rules**

```
#####  
#                               CAARF Rules                               #  
#####  
  
# Regra de assinatura para identificação de ataques do tipo DoS e DDoS  
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"Possible TCP Syn Flood Attck DoS  
port 80"; flags:S; flow:to_server; threshold: type both,  
track by_src, count 40, seconds 5; classtype:attempted-dos; sid:10001; ver:1;)  
  
#Regra de assinatura para identificação de ataques do tipo ICMP Flood Smurf  
alert icmp any any -> any any (msg:"Possible Smurf Attack";  
sid:1000003; itype:8;)  
  
#####
```

APÊNDICE B – CONFIGURAÇÃO BÁSICA DO SURICATA IDS

Código Suricata.yaml

```
#####
#                               Suricata.yaml                               #
#####

%YAML 1.1
---

# Suricata configuration file. In addition to the comments describing all
# options in this file, full documentation can be found at:
# https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Suricatayaml

##
## Step 1: inform Suricata about your network
##

vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[172.16.0.0/24]"

    EXTERNAL_NET: "!$HOME_NET"
    #EXTERNAL_NET: "any"

    HTTP_SERVERS: "$HOME_NET"
    SMTP_SERVERS: "$HOME_NET"
    SQL_SERVERS: "$HOME_NET"
    DNS_SERVERS: "$HOME_NET"
    TELNET_SERVERS: "$HOME_NET"
    AIM_SERVERS: "$EXTERNAL_NET"
    DNP3_SERVER: "$HOME_NET"
    DNP3_CLIENT: "$HOME_NET"
    MODBUS_CLIENT: "$HOME_NET"
    MODBUS_SERVER: "$HOME_NET"
    ENIP_CLIENT: "$HOME_NET"
    ENIP_SERVER: "$HOME_NET"

  port-groups:
```

```
HTTP_PORTS: "80"
SHELLCODE_PORTS: "!80"
ORACLE_PORTS: 1521
SSH_PORTS: 22
DNP3_PORTS: 20000
MODBUS_PORTS: 502
FILE_DATA_PORTS: "[$HTTP_PORTS,110,143]"
FTP_PORTS: 21

##
## Step 2: select the rules to enable or disable
##
default-rule-path: /etc/suricata/rules
rule-files:
- caarf.rules
- botcc.rules
- http-events.rules # available in suricata sources under rules dir

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config

##
## Step 3: select outputs to enable
##
# The default logging directory. Any log or output file will be
# placed here if its not specified with a full path name. This can be
# overridden with the -l command line parameter.
default-log-dir: /var/log/suricata/

# Configure the type of alert (and other) logging you would like.
outputs:
# a line based alerts log similar to Snort's fast.log
- fast:
  enabled: yes
  filename: fast.log
  append: yes
  filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'
```

APÊNDICE C – CONFIGURAÇÃO DO GUARDIAN

Código Guargian.conf

```
#####
#                               Guardian.conf                               #
#####
# The machines IP address that is visable to the internet
# If this is left undefined, then guardian will attempt to get the information
# from ifconfig, as long as it has an interface to use. This would be useful
# HostIpAddr

# The IP address of remote controller which will be requested to
# block/unblock hosts
RemoteController 172.16.0.1

# Here we define the interface which we will use to guess the IP address, and
# block incoming offending packets. This is the only option that is required
Interface ens33

# The last octet of the ip address, which gives us the gateway address.
HostGatewayByte 1

# Guardian's log file
LogFile /var/log/guardian.log

# Snort's alert file. This can be the snort.alert file, or a syslog file
# There might be some snort alerts that get logged to syslog which guardian
AlertFile /var/log/suricata/fast.log

# The list of ip addresses to ignore
IgnoreFile /etc/guardian.ignore

# This is a list of IP addresses on the current host, in case there is more
# than one. If this file doesn't exist, then it will assume you want to run
TargetFile /etc/guardian.target

# The time in seconds to keep a host blocked. If undefined, it defaults to
# 99999999, which basicly disables the feature.
TimeLimit 86400
```


APÊNDICE D – CONFIGURAÇÃO DE INTEGRAÇÃO LOGSTACK

Código Logstash.conf

```
#####
#                               Logstash.conf                               #
#####

input {
  file {
    path => ["/var/log/suricata/eve.json"]
    codec => "json"
    type => "SuricataIDPS"
  }
}

filter {
  if [type] == "SuricataIDPS" {
    date {
      match => [ "timestamp", "ISO8601" ]
    }
    ruby {
      code => "
        if event.get('[event_type]') == 'fileinfo'
          event.set('[fileinfo][type]', event.get('[fileinfo][magic]').to_s.split(',')[0])
        end
      "
    }
    ruby{
      code => "
        if event.get('[event_type]') == 'alert'
          sp = event.get('[alert][signature]').to_s.split(' group ')
          if (sp.length == 2) and /\A\d+\z/.match(sp[1])
            event.set('[alert][signature]', sp[0])
          end
        end
      "
    }
  }
}

if [src_ip] {
  geoip {
    source => "src_ip"
    target => "geoip"
    #database => "/opt/logstash/vendor/geoip/GeoLiteCity.dat"
    add_field => [ "[geoip][coordinates]", "% {[geoip][longitude]}" ]
    add_field => [ "[geoip][coordinates]", "% {[geoip][latitude]}" ]
  }
}
```

```
mutate {
  convert => [ "[geoiip][coordinates]", "float" ]
}
if ![geoiip.ip] {
  if [dest_ip] {
    geoiip {
      source => "dest_ip"
      target => "geoiip"
      #database => "/opt/logstash/vendor/geoiip/GeoLiteCity.dat"
      add_field => [ "[geoiip][coordinates]", "% {[geoiip][longitude]}" ]
      add_field => [ "[geoiip][coordinates]", "% {[geoiip][latitude]}" ]
    }
    mutate {
      convert => [ "[geoiip][coordinates]", "float" ]
    }
  }
}
}
```

APÊNDICE E – CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 1**Código Atomix.conf**

```
#####  
#                               Atomix 1                               #  
#####  
cluster {  
  cluster-id: onos  
  node {  
    id: caarf-atomix1  
    address: "172.16.0.1:5679"  
  }  
  discovery {  
    type: bootstrap  
    nodes.1 {  
      id: caarf-atomix1  
      address: "172.16.0.1:5679"  
    }  
    nodes.2 {  
      id: caarf-atomix2  
      address: "172.16.0.2:5679"  
    }  
    nodes.3 {  
      id: caarf-atomix3  
      address: "172.16.0.3:5679"  
    }  
  }  
}management-group {  
  type: raft  
  partitions: 1  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}  
partition-groups.raft {  
  type: raft  
  partitions: 3  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}
```

APÊNDICE F – CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 2**Código Atomix.conf**

```
#####  
#                               Atomix 2                               #  
#####  
cluster {  
  cluster-id: onos  
  node {  
    id: caarf-atomix1  
    address: "172.16.0.1:5679"  
  }  
  discovery {  
    type: bootstrap  
    nodes.1 {  
      id: caarf-atomix1  
      address: "172.16.0.1:5679"  
    }  
    nodes.2 {  
      id: caarf-atomix2  
      address: "172.16.0.2:5679"  
    }  
    nodes.3 {  
      id: caarf-atomix3  
      address: "172.16.0.3:5679"  
    }  
  }  
}management-group {  
  type: raft  
  partitions: 1  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}  
partition-groups.raft {  
  type: raft  
  partitions: 3  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}
```

APÊNDICE G – CONFIGURAÇÃO DO ATOMIX INSTÂNCIA 3**Código Atomix.conf**

```
#####  
#                               Atomix 3                               #  
#####  
cluster {  
  cluster-id: onos  
  node {  
    id: caarf-atomix1  
    address: "172.16.0.1:5679"  
  }  
  discovery {  
    type: bootstrap  
    nodes.1 {  
      id: caarf-atomix1  
      address: "172.16.0.1:5679"  
    }  
    nodes.2 {  
      id: caarf-atomix2  
      address: "172.16.0.2:5679"  
    }  
    nodes.3 {  
      id: caarf-atomix3  
      address: "172.16.0.3:5679"  
    }  
  }  
}management-group {  
  type: raft  
  partitions: 1  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}  
partition-groups.raft {  
  type: raft  
  partitions: 3  
  storage.level: disk  
  members: [caarf-atomix1, caarf-atomix2, caarf-atomix3]  
}
```

APÊNDICE H – CONFIGURAÇÃO DO ONOS INSTÂNCIA 1**Código Cluster.json**

```
#####  
#                               Cluster onos1                               #  
#####  
{  
  "name": "onos",  
  "node": {  
    "id": "caarf-onos1",  
    "ip": "172.16.0.10",  
    "port": 9876  
  },  
  "storage": [  
    {  
      "id": "caarf-atomix1",  
      "host": "caarf-atomix1",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix2",  
      "host": "caarf-atomix2",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix3",  
      "host": "caarf-atomix3",  
      "port": 5679  
    }  
  ]  
}  
##### fim #####
```

APÊNDICE I – CONFIGURAÇÃO DO ONOS INSTÂNCIA 2

Código Cluster.json

```
#####  
#                               Cluster onos2                               #  
#####  
{  
  "name": "onos",  
  "node": {  
    "id": "caarf-onos2",  
    "ip": "172.16.0.20",  
    "port": 9876  
  },  
  "storage": [  
    {  
      "id": "caarf-atomix1",  
      "host": "caarf-atomix1",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix2",  
      "host": "caarf-atomix2",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix3",  
      "host": "caarf-atomix3",  
      "port": 5679  
    }  
  ]  
}  
  
##### fim #####
```

APÊNDICE J – CONFIGURAÇÃO DO ONOS INSTÂNCIA 3**Código Cluster.json**

```
#####  
#                               Cluster onos3                               #  
#####  
{  
  "name": "onos",  
  "node": {  
    "id": "caarf-onos3",  
    "ip": "172.16.0.30",  
    "port": 9876  
  },  
  "storage": [  
    {  
      "id": "caarf-atomix1",  
      "host": "caarf-atomix1",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix2",  
      "host": "caarf-atomix2",  
      "port": 5679  
    },  
    {  
      "id": "caarf-atomix3",  
      "host": "caarf-atomix3",  
      "port": 5679  
    }  
  ]  
}  
  
##### fim #####
```