UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente

Dissertação de Mestrado

Bruno Nunes Barreto

São Cristóvão – Sergipe

2020

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Bruno Nunes Barreto

Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Orientador(a): Admilson de Ribamar Lima Ribeiro

São Cristóvão – Sergipe

2020

Bruno Nunes Barreto
Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente/ Bruno Nunes Barreto. – São Cristóvão – Sergipe, 2020-
82 p. : il. (algumas color.) ; 30 cm.

Orientador: Admilson de Ribamar Lima Ribeiro

Dissertação de Mestrado – UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO, 2020.

1. Fog Computing. 2. Cloud Computing. 3. QoS. 4. SLA. 5. Gerenciamento de Recurso I. Admilson de Ribamar Lima Ribeiro. II. Universidade Federal de Sergipe. III. Centro de Ciências Exatas e Tecnologia. IV. Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente

CDU 02:141:005.7

Bruno Nunes Barreto

Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Trabalho aprovado. São Cristóvão – Sergipe, 13 de Agosto de 2020:

Admilson de Ribamar Lima Ribeiro
Orientador

Edward David Moreno Ordonez
Interno a instituição

José Augusto Andrade Filho
Externo a instituição

São Cristóvão – Sergipe
2020

UNIVERSIDADE FEDERAL DE SERGIPE
PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA
COORDENAÇÃO DE PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ata da Sessão Solene de Defesa da Dissertação do Curso de Mestrado em Ciência da Computação-UFS. Candidato: Bruno Nunes Barreto

Em 13 dias do mês de agosto do ano de dois mil e vinte, com início às 14h00min, realizou-se na Sala virtual https://meet.jit.si/UFSdefesaPROCC.BrunoNB. A Sessão Pública de Defesa de Dissertação de Mestrado do candidato **Bruno Nunes Barreto**, que desenvolveu o trabalho intitulado: **"** *Uma Arquitetura de Fog Computing Virtualizada para Prover Gerenciamento de Recursos, QoS e SLA em um Ambiente Inteligente***",** sob a orientação do Prof. Dr. **Admilson De Ribamar Lima Ribeiro**. A Sessão foi presidida pelo Prof. Dr. **Admilson De Ribamar Lima Ribeiro** (PROCC/UFS), que após a apresentação da dissertação passou a palavra aos outros membros da Banca Examinadora, Prof. Dr. **Edward David Moreno Ordonez** (PROCC/UFS) e, em seguida, ao Prof. Dr. **José Augusto Andrade Filho** (IFS - Instituto Federal de Sergipe). Após as discussões, a Banca Examinadora reuniu-se e considerou o mestrando (a) Aprovado "*(aprovado/reprovado)"*. Atendidas as exigências da Instrução Normativa 01/2017/PROCC, do Regimento Interno do PROCC (Resolução 67/2014/CONEPE), Resolução nº 25/2014/CONEPE e da Portaria nº 413 de 27 de maio de 2020 (Banca por videoconferência) que regulamentam a Apresentação e Defesa de Dissertação, e nada mais havendo a tratar, a Banca Examinadora elaborou esta Ata que será assinada pelos seus membros e pelo mestrando.

Cidade Universitária "*Prof. José Aloísio de Campos",* 13 de agosto de 2020.
"participação à distância por videoconferência"

Prof. Dr.Admilson De Ribamar Lima Ribeiro
(PROCC/UFS)
Presidente

Prof. Dr. Edward David Moreno Ordonez
(PROCC/UFS)
Examinador Interno

Prof. Dr. José Augusto Andrade Filho
(IFS)
Examinador Externo

Bruno Nunes Barreto
Candidato

“Dedico este trabalho a minha família que está sempre ao meu lado, seja qual for o momento, me ajudando de todas as formas.”

Agradecimentos

Agradeço a Deus, a quem sempre recorro nos momentos mais difíceis, por tudo que tenho e pela minha vida.

Meus sinceros agradecimentos ao meu professor, orientador e tio, Admilson Ribeiro, que mesmo sabendo de minhas dificuldades me incentivou, orientou, e me passou seus conhecimentos. Aos professores Edward (UFS) e José Augusto (IFS), por darem suas significativas contribuições e aceitarem o convite de compor a banca.

A minha esposa Catarina, pela compreensão e incentivo constante nas minhas decisões e principalmente, por partilhar do seu amor e amizade durante todos estes anos.

Aos meus pais, Beto e Rosa, pelo amor incondicional e por tudo que fazem por mim para me tornar capaz de buscar as minhas próprias conquistas. Aos meus irmãos Sérgio e Theo, que tenho plena certeza que posso contar a qualquer hora, pela amizade, apoio e confiança. Às minhas sobrinhas Milena e Fernanda por existirem em minha vida. Aos demais familiares pelos momentos que compartilhamos.

A minha tia Kátia, que me incentivou a seguir este rumo. Ao meu tio Renoir pelas correções gramaticais, conversas e incentivos em busca não apenas deste título, mas em minhas decisões. Ao meu primo Dija pela parceria e aventuras que desafogavam o meu estresse.

Aos meus filhos de quatro patas (Natan, Bétany, Lina e Sabrina), pelo amor e por garantirem minha diversão em todos os momentos que estamos juntos.

Estendo a minha gratidão aos amigos e colegas Alexandre, Felipe Faustino, Hugo, Felipe (Japa), Alef, Alana, Thiago, Mislene, Cícero, Thauane e Ademir pelas contribuições de grande valor, apoio técnico e companheirismo durante o período do mestrado.

Aos demais professores e funcionários dos departamentos de computação (PROCC/DCOMP) e inglês sem fronteira da UFS, em especial a Elaine e Adinésia, que me auxiliaram nas questões burocráticas da UFS. Agradeço também a todos que contribuíram diretamente e indiretamente para que eu chegasse até aqui.

Por fim, ao Programa de Pós-Graduação em Ciência da Computação da UFS, à Universidade Federal de Sergipe (UFS), à FAPITEC/SE e à CAPES pela oportunidade a mim concedida de buscar e contribuir com o conhecimento.

“A tarefa não é tanto ver aquilo que ninguém viu,
mas pensar o que ninguém ainda pensou
sobre aquilo que todo mundo vê.”
(Arthur Schopenhauer)

Resumo

A *Fog Computing* representa uma mudança no uso e na arquitetura da Internet das coisas combinada com a computação em nuvem, onde dispositivos ubíquos com diferentes níveis de autonomia e necessidades passam por *gateways* inteligentes capazes de fornecer recursos e serviços flexíveis para os usuários finais na borda da rede. Em função da amplitude proposta e heterogeneidade de tecnologias e "objetos" na Internet das coisas, sua interligação direta com a nuvem torna-se impraticável para alguns tipos de aplicações, dentre elas, algumas voltadas para ambientes inteligentes, fazendo necessária a interferência de um novo paradigma capaz de suprir tais necessidades. Dessa forma, motivada pela quantidade de questões em aberto e por ser um tema atual, esta dissertação demonstra a capacidade da arquitetura de *fog computing* virtualizada (AFCV), desenvolvida e implementada, para prover o gerenciamento de recursos, QoS e auxiliar contratos de acordo de nível de serviço (SLA). Os experimentos realizados em ambiente real, validam as vantagens da AFCV em relação à nuvem convencional por meio de resultados como a eficiência na comunicação (onde a AFCV superou a nuvem em 52,97% com o protocolo UDP e em 79,72% o TCP), largura de banda, entre outros. Esta dissertação preenche lacunas relacionadas à *fog computing* e possibilita a garantia e o fornecimento de serviços capazes de serem implementados em ambientes inteligentes (como por exemplo o de *smart home*), bem como auxilia o desenvolvimento de modelos de negócio voltados para a mesma.

Palavras-chave: Fog Computing, Computação em Nuvem, QoS, SLA, Gerenciamento de Recurso.

Abstract

Fog Computing represents a change in the use and architecture of the Internet of Things combined with Cloud Computing, where ubiquitous devices with different levels of autonomy and needs go through intelligent gateways capable of providing flexible resources and services to end-users at the edge of the network. Due to the proposed breadth and heterogeneity of technologies and "objects" in the Internet of things, their direct interconnection with the cloud becomes impractical for some types of applications, among them, some aimed at intelligent environments, making it necessary the interference of a new paradigm capable of meeting such needs. Thus, motivated by the number of open questions and because it is a current topic, this research work demonstrates the capacity of the virtualized fog computing architecture (VFCA), developed and implemented, to provide resource management, QoS, and support to contracts of service level agreement (SLA). The experiments carried out in a real environment, validate the advantages of VFCA concerning the conventional cloud through results such as efficiency in communication (where AFCV surpassed the cloud by 52.97% with the UDP protocol and 79.72% with TCP), bandwidth, among others. This research work fills gaps related to fog computing and makes it possible to guarantee and provide services capable of being implemented in smart environments (such smart home), as well as helping to develop business models aimed at it.

Keywords: Fog Computing, Cloud Computing, QoS, SLA, Resource Management.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

QoS	Quality of Service
SLA	Service Level Agreement
IoT	Internet of Things
VNF	Virtual Network Functions
F2C	fog-to-cloud
OF2C	Optimized Fog-to-Cloud
IaaS	Infrastructure as a service
PaaS	Plataform as a service
STS	Smart Transportation Safety
ICN	Information Centric Networking
CDN	Content Delivery Network
VM	Virtual Machine
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
SCP	Secure Copy Protocol
DNS	Domain Name System
AFCV	Arquitetura de Fog Computing Virtualizada

Sumário

1 Introdução

1.1 Apresentação Geral

A evolução tecnológica de equipamentos embarcados tem possibilitado a comunicação virtual com determinados objetos para que possamos gerenciá-los e operá-los a distância através da Internet. Com a finalidade de aumentar cada vez mais a capacidade interacional dos sistemas, um novo paradigma denominado genericamente por IoT (*Internet of Things*) vem emergindo, (ATZORI; IERA; MORABITO, 2010). Por meio da integração das mais variadas tecnologias, visa possibilitar a comunicação em rede entre pessoas, objetos e coisas com diferentes níveis de autonomia, extraindo e/ou fornecendo serviços e informações entre si ou para outros dispositivos através da Internet.

A arquitetura de IoT pode ser tratada como um sistema físico, virtual ou híbrido, podendo fazer uso de tecnologias como a computação em nuvem (RAY, 2016), capaz de suprir as limitações de computação e armazenamento em dispositivos inteligentes, além de fornecer recursos elásticos aos mesmos (YI; LI; LI, 2015). Segundo Yi, Li e Li (2015), Bonomi et al. (2012) e Mahmud, Kotagiri e Buyya (2016), diante da exigência de suporte à mobilidade, distribuição geográfica, reconhecimento de localização e demanda por baixa latência de algumas aplicações, a nuvem encontra algumas dificuldades.

Para suplantar essas dificuldades, características da nuvem foram trazidas para a borda da rede (SHARMA; CHEN; PARK, 2018), (AI; PENG; ZHANG, 2018) e (VOHRA; DAVE, 2018), formando assim a *fog computing*, ou simplesmente *fog*, que como um elo entre a IoT e a nuvem, induz as funcionalidades extras necessárias para o processamento específico de aplicativos, como filtragem e agregação, antes de transferir os dados para a nuvem (AL-DOGHMAN et al., 2016).

Aproveitando a capacidade da IoT, uma grande diversidade de soluções e aplicações inteligentes para as mais diversas áreas, como *smart city* e *smart home*, vêm sendo propostas e exigindo cada vez mais da mesma, porém, a aplicabilidade da *fog* se estende a outras áreas. No

entanto, esse avanço de geração tem se apresentado de forma altamente complexa, o que vem demandando e movimentando pesquisadores das mais variadas áreas do conhecimento, além da necessidade de criação de ambientes para a análise de desempenho destes estudos. Alguns desafios da *fog* são listados por (MAHMUD; KOTAGIRI; BUYYA, 2016) e (MOURADIAN et al., 2018), que consideram a importância de identificar técnicas e métricas apropriadas para o provisionamento e gerenciamento eficiente de recursos. Li, Li e Zhao (2014), afirmam que grande número de *links* e interações diferentes entre nós de borda na IoT faz dela um sistema complexo e escalável. Isso traz dificuldades para satisfazer os requisitos dinâmicos de qualidade de serviços (*quality of service* - QoS). (PRODAN; OSTERMANN, 2009) e (MOURADIAN et al., 2018) defendem que a ausência de gerência de acordo de nível de serviço (*service level agreement* - SLA), bem como de métricas sustentáveis, dificultam a manutenção de um QoS aceitável em ambientes altamente dinâmicos.

1.2 Motivação

A IoT contribui com o rápido crescimento e diversidade de dispositivos conectados à Internet, que segundo Reinsel, Gantz e Rydning (2018) produzirão mais de 90 zettabytes dos dados mundiais em 2025. No entanto, a Internet não é eficiente nem escalonável o suficiente para lidar com essas enormes quantidades de dados da IoT, bem como a transferência de dados importantes é cara, consumindo uma enorme quantidade de largura de banda, tempo e energia (SHARMA; CHEN; PARK, 2018). Isso sugere a *fog computing* como possível arquitetura a suportar essa demanda (AL-DOGHMAN et al., 2016).

Segundo Mouradian et al. (2018), devido às pesquisas sobre a *fog computing* estarem ainda em estágio inicial, apesar da existência do dispositivo comercial CISCO IOx, nenhum caso de uso ou implementação no mundo real está disponível publicamente. Com isso, torna-se necessária a realização de uma análise de desempenho em cima dos novos modelos propostos. Além disso vários outros desafios relacionados a arquitetura, teste e padronização, governança e colaboração são citados por (VERMESAN; FRIESS; FURNESS, 2012), incluindo a necessidade de desenvolvimento e aperfeiçoamento de estruturas físicas e lógicas, levantamento de requisitos, mecanismos que cubram aspectos legais, responsabilidades e que garantam a QoS e a interoperabilidade entre provedores. (TECKELMANN; REICH; SULISTIO, 2011) e (LI; LI; ZHAO, 2014) também destacam obstáculos relacionados ao SLA, QoS e limitação de energia. (PRODAN; OSTERMANN, 2009) conclui que o fornecimento de SLA, soluções de *middleware* abertas e interoperáveis, e métricas de desempenho sustentadas para aplicativos de computação de alto desempenho, são três elementos com a maior necessidade de pesquisas futuras.

Ratificando os problemas a serem enfrentados, (MOURADIAN et al., 2018) afirmam que o gerenciamento de SLA e a análise de métricas como largura de banda de *uplink* e *downlink* estão relacionados com a QoS, permanecendo em aberto para consideração principalmente em

ambientes inteligentes, onde segundo os mesmos, nenhuma publicação atende a esse critério.

1.3 Objetivos

O objetivo principal deste trabalho de pesquisa é o desenvolvimento e implementação de uma arquitetura de *fog computing* virtualizada para prover gerenciamento de recursos, QoS e auxílio para SLA.

1.3.1 Objetivos Específicos:

Para que o objetivo principal seja alcançado, os objetivos específicos também devem ser concluídos, são eles:

• Minimização do problema referente a latência através da arquitetura de *fog computing*.

• Gerenciamento dos recursos oferecidos pela *fog*.

• Fornecimento de parâmetros configuráveis para a comercialização de serviços de *fog computing* através de contratos de SLA.

• Garantia da qualidade de serviço através do estabelecimento das métricas apropriadas.

• Gerenciamento da infraestrutura distribuída e QoS através de ferramentas apropriadas.

• Fornecimento da confiabilidade do serviço através dos resultados das análises de desempenho.

1.4 Justificativa

Publicações recentes, motivadas pelo que foi considerado na seção 1.2, descrevem num âmbito geral a *fog computing* (MOURADIAN et al., 2018; YI; LI; LI, 2015; AL-DOGHMAN et al., 2016), abordam questões específicas como o acordo de nível de serviço e qualidade de serviço (LI; LI; ZHAO, 2014; TECKELMANN; REICH; SULISTIO, 2011; PRODAN; OSTERMANN, 2009), e soluções com análise de desempenho em ambientes de *smart city* e *smart home* (CARMO et al., 2017; MASIP-BRUIN et al., 2016).

Conforme Mouradian et al. (2018), a *fog* oferece vantagens adicionais, como a baixa latência e processamento em locais específicos próximos dos dispositivos. Neste trabalho, os autores estabelecem critérios e analisam estudos primários existentes na perspectiva arquitetônica e algorítmica, sumarizando através de tabelas as contribuições e questões em aberto de cada uma delas, sugerindo em seguida possíveis soluções para cada problema, dentre elas, a utilização de abordagens de SLA de gerenciamento de nuvem para ambientes de *fog*. Por fim diante das análises realizadas, é entendido que tanto as medições experimentais quanto as simuladas confirmam a possibilidade de redução da latência, o que é importante para aplicativos em tempo

real. Já Li, Li e Zhao (2014), propõe um modelo de agendamento de QoS em três camadas orientado para serviços da IoT, ambiente ao qual os autores relacionam mais atributos de QoS, como precisão de informações, recursos de rede e consumo de energia necessários, e a ampla cobertura da IoT. Carmo et al. (2017) considerou o compartilhamento eficiente de recursos de rede do cliente por meio da criação de fatias de rede configuradas através de *software defined networking* (SDN) e *virtual network functions* (VNF) para trazer serviços personalizados mais perto de dispositivos móveis e sensores, considerando um ambiente de *smart city*, analisando os resultados, e comparando o atraso médio da nuvem que foi de aproximadamente 133ms contra 12 e 5,3ms referentes a dois nós da *fog*.

Apesar das publicações encontradas apresentarem soluções para algumas questões, é perceptível a existência de várias outras ainda em aberto (MOURADIAN et al., 2018; YI; LI; LI, 2015), demandando avaliações e experimentos, como as que envolvem definição de SLA e métricas para fornecimento de QoS em ambientes de *fog computing* que também carecem de gerenciamento.

1.5 Método de Pesquisa

Este trabalho de pesquisa foi classificado de acordo com (SILVA; MENEZES, 2005), tendo suas características descritas da seguinte forma:

• Quanto à natureza: aplicada, pois tem o objetivo de gerar conhecimento necessário para aplicação prática de uma arquitetura de *fog computing* a fim de resolver problemas e atender a interesses comerciais e científicos.

• Quanto à abordagem do problema: qualitativa devido ao uso do processo de mapeamento sistemático, através do qual foram selecionados os estudos relevantes ao tema abordado neste trabalho, a fim de fornecer um melhor embasamento teórico, bem como a identificação de algumas lacunas exploradas pelo mesmo. Essa abordagem é também quantitativa, pois foram definidas métricas pertinentes ao ambiente de *fog computing* e realizadas medições para avaliar a arquitetura desenvolvida.

• Quanto aos objetivos: exploratórios e descritivos, buscando uma maior familiarização com o problema através de pesquisas bibliográficas e exploração dos recursos existentes e mais utilizados pelas soluções propostas nos trabalhos relacionados, transformando-os numa arquitetura de *fog computing* para disponibilizar serviços com qualidade e mensuráveis para a população, através de técnicas padronizadas de coleta de dados.

• Quanto aos procedimentos técnicos: trata-se de uma pesquisa bibliográfica, devido a utilização de materiais já publicados e disponíveis através da Internet. Esse levantamento teórico foi fundamental para o desenho da arquitetura que é o principal objetivo deste trabalho. Foi cuidadosamente implementada considerando as tecnologias, *softwares* e ferramentas que

apresentaram melhores resultados e são capazes de atender através de seus requisitos às demandas de gerenciamento, QoS e SLA. Ela também possui caráter experimental, pois foram definidas métricas, selecionadas suas variáveis e observado seus efeitos na arquitetura. Foi realizada uma avaliação de desempenho da arquitetura e seus componentes em várias situações, incluindo a utilização de serviço de *streaming* de vídeo através do *software* Plex (PLEX, 2019) para a geração de carga no ambiente a fim de identificar possíveis limitações e benefícios impostos pela mesma. Métricas como capacidade da rede, *jitter*, largura de banda de *uplink* e *downlink*, utilização de CPU e memória, são avaliadas e comparadas entre os dispositivos e entre a nuvem e a *fog*. Para isso foram utilizadas ferramentas como o portainer (PORTAINER, 2018), Nginx (NGINX, 2019), iperf (IPERF, 2019) e BlazeMeter (BLAZEMETER, 2019), instaladas em um ambiente clusterizado e implementado com a ferramenta *Docker container* (DOCKER, 2018).

1.6 Estrutura do Documento

Este documento está estruturado em seis capítulos. No capítulo 1 é realizada uma introdução contemplando uma apresentação geral que contextualiza o tema de forma resumida. Apresenta sua problemática, a motivação para a realização desta pesquisa em relação com a hipótese de trabalho, os objetivos a serem atingidos, a justificativa para a escolha do tema e o método de pesquisa adotado.

No capítulo 2 é apresentada a fundamentação teórica sobre a *fog computing*, IoT, computação em nuvem, QoS, SLA, e gerenciamento de recursos. Os trabalhos relacionados são abordados no capítulo 3 onde também é explicado o processo de pesquisa e apresentada tabela comparativa contendo as características dos mesmos, bem como deste trabalho de pesquisa. No capítulo 4 é apresentada a proposta desta dissertação, na qual descrevemos a arquitetura desenvolvida (AFCV), assim como sua composição, características, funcionalidades e a prototipação realizada para a coleta de resultados.

Tais resultados são apresentados no capítulo 5, e as conclusões contendo os trabalhos futuros e as publicações desenvolvidas até aqui aparecem no capítulo 6.

2

Fundamentação Teórica

Neste capítulo são apresentados alguns conceitos da *fog computing*, sua relação com as visões e requisitos da IoT, além de um comparativo com a nuvem e algumas definições e características sobre o gerenciamento de recursos, QoS e SLA.

2.1 Conceito da Fog Computing

Há muitos desafios que devem ser enfrentados na arquitetura de rede tradicional devido ao rápido crescimento da diversidade e do número de dispositivos conectados à Internet (SHARMA; CHEN; PARK, 2018).

Para suportar a demanda computacional de aplicativos sensíveis a latência, foi introduzido um novo paradigma da computação chamado *fog Computing*. Geralmente, a *fog* fica mais próxima dos dispositivos localizados na borda da rede e estende as facilidades de computação, armazenamento e rede baseadas na nuvem (MAHMUD; KOTAGIRI; BUYYA, 2016).

A *fog* atua como um elo entre o IoT e a nuvem, introduzindo uma camada intermediária, capaz de analisar os dados e tomar decisões antes mesmo de transferi-los para a nuvem. Deve ser capaz de decidir o que deve ser enviado (o conteúdo), como (formato de dados) e quando enviar (tempo). Durante esse processo, também precisa excluir alguns dados redundantes ou inválidos e agregar os dados complementares nas dimensões de espaço e tempo (AL-DOGHMAN et al., 2016).

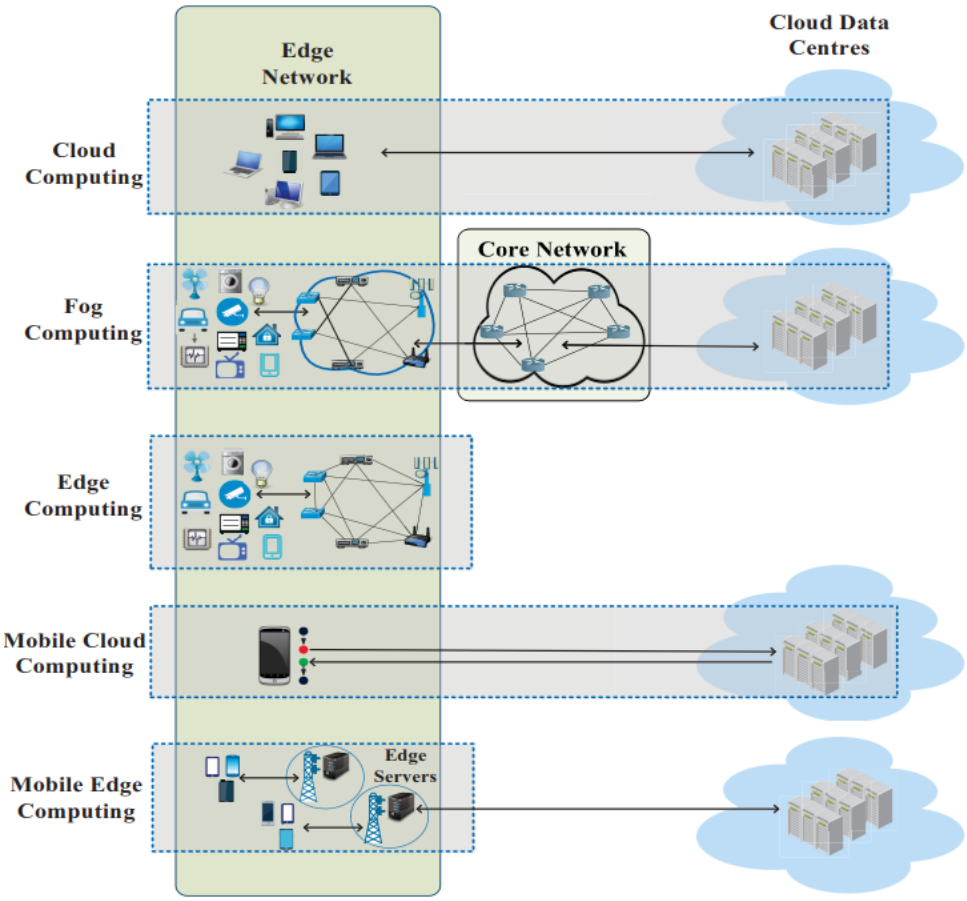
Algumas de suas características citadas por Bonomi et al. (2012) são: heterogeneidade, distribuição geográfica, proximidade da borda, percepção de localização e baixa latência, interação em tempo real, suporte à mobilidade, grande escala de redes de sensores prevalentes para acesso sem fio e interoperabilidade.

Dos muitos modelos orientados à computação e *software* que estão sendo adotados, a

fog computing conquistou um amplo grupo de pesquisadores e a indústria, mas ainda há muita confusão sobre sua definição, posição, função e aplicação precisa. Como exemplo, temos o trabalho de Yi, Li e Li (2015) que cita a *edge computing* como sinônimo da *fog* e os artigos de Ai, Peng e Zhang (2018) e Mahmud, Kotagiri e Buyya (2016), que definem a arquitetura da *edge* sem a utilização da nuvem. Ou seja, ao invés das três camadas existentes na *fog* ela possui apenas duas, conforme pode ser verificado na figura 1. A IoT exige resposta em tempo real para muitos aplicativos e serviços, e isso torna a *fog Computing* uma plataforma adequada para alcançar objetivos de autonomia e eficiência (AL-DOGHMAN et al., 2016).

Embora a *fog* seja reconhecida como o modelo de computação mais apropriado para a IoT, uma das principais razões dela não ter sido amplamente utilizada reside na necessidade de substituição do *firmware* e *hardware* dos equipamentos de rede. Tarefa esta que não é clara para o operador encarregado de executá-la, implicando então uma curva de aprendizado para a realização da mesma. Como ela é baseada na colaboração entre vários operadores de infraestrutura e provedores de serviços, não está claro quem opera e gerencia a infraestrutura (KIM; CHUNG, 2018).

Figura 1 – Arquiteturas.

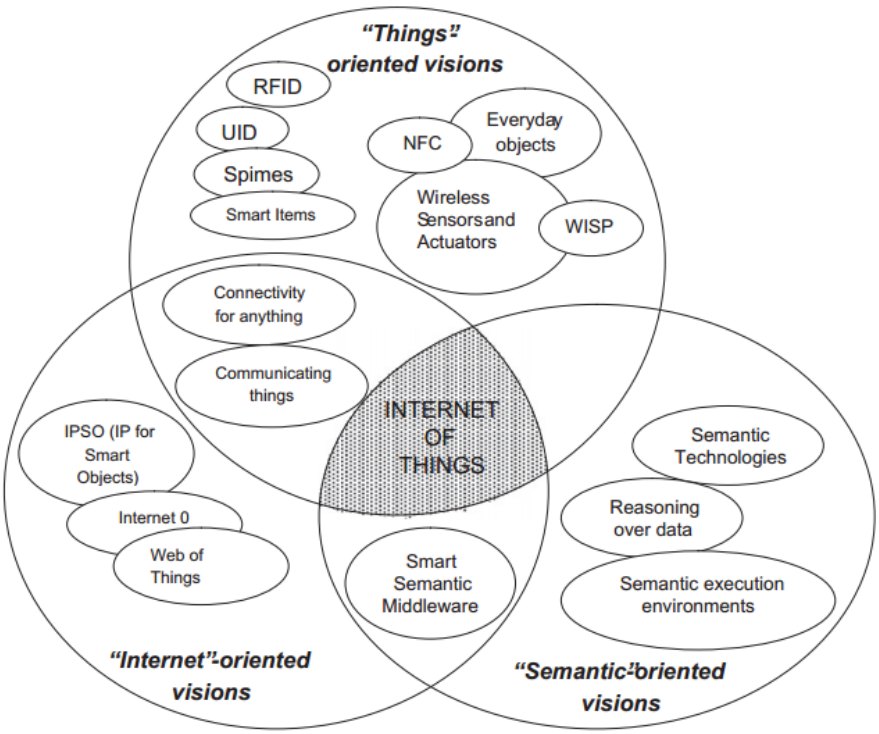


Fonte: Mahmud, Kotagiri e Buyya (2016).

2.2 Fog Computing e a Internet of Things

A IoT abre caminho para diversas áreas que se aproveitam de suas tecnologias, como a *smart city* (MADAKAM; RAMASWAMY, 2015) e a *smart home* (CHEN; AZHARI; LEU, 2018), proporcionando diferentes visões de pesquisa deste paradigma. Visão de rede ou infraestrutura, de objetos ou coisas e visão semântica ou de informação (ATZORI; IERA; MORABITO, 2010).

Figura 2 – Visões da IoT.



Fonte: Atzori, Iera e Morabito (2010).

Na visão orientada à Internet, o foco é concentrado na infraestrutura formada por estes novos elementos em suas diferentes combinações, no reflexo desta nova organização na infraestrutura da Internet como um todo, nos serviços que executam sobre esta plataforma e na quantidade, tipo e controles sobre este novo padrão de tráfego gerado. Baseado nessas características, podemos observar que a *fog* e a nuvem podem fazer parte desta arquitetura.

A visão das "coisas" (*things*) analisa os objetos, seus possíveis níveis de autonomia e tecnologias necessárias para a produção destes *"smart items"*. Os "nós" (ou objetos) da *fog* estão situados numa camada próximo a borda da rede onde esses dispositivos de IoT se encontram. Estes nós podem ser dispositivos com poucos recursos, como pontos de acesso, roteadores, *switches*, estações base e dispositivos finais, ou máquinas ricas em recursos, como *cloudlet* e o IOx da Cisco, pois ambos podem fornecer recursos para serviços (YI; LI; LI, 2015) e (MAHMUD; KOTAGIRI; BUYYA, 2016).

Já a visão semântica trata as pesquisas focadas na quantidade, tipo e importância das

informações produzidas. Como localizar, utilizar, manipular e armazenar tais informações de forma inteligente e eficiente. Analisando essas visões, conseguimos resumir o modelo da IoT em produto, informação e comunicação. Onde o produto seria o sistema por completo, a informação é aquilo que é transmitido e a comunicação seria como a informação é transmitida. Esta visão está relacionada na *fog computing* com as questões referidas à tomada de decisão que definem o local onde os dados serão tratados, sejam nos nós da *fog* ou enviados para a nuvem (AL-DOGHMAN et al., 2016).

Apesar dessas três visões da IoT bem definidas, essa compilação coloca em jogo várias questões de requisitos citadas por (CLUSTER, 2012). São elas:

Arquitetura: desenvolver e aperfeiçoar as estruturas física e lógica, como componentes e software respectivamente, incluindo questões de identificação do objeto, virtualização e descentralização, além de garantir a interoperabilidade entre os setores de aplicação.

Segurança e Confiança: desenvolver projetos com mecanismos e estruturas que garantam a confiança e o controle dos usuários, e contextos particulares sobre as informações das aplicações e do ciclo de vida.

Software e plataformas de *middleware*: suportar dispositivos de detecção e instâncias de objetos dispostos na IoT para análise e processamento de fluxos de dados, filtragem de eventos e gestão de complexidade.

Interfaces: integrar diferentes interfaces no sentido de melhorar a interação homem-máquina, aumentando a experiência do usuário na obtenção de informação.

Sensores inteligentes: integrar os sensores à capacidade de auto adaptação às mudanças sofridas pelo ambiente e aos dispositivos de captação de energia.

Testes e Padronização: assim como este trabalho de dissertação que levanta os requisitos da *fog computing*, deve haver o levantamento de requisitos nas outras áreas envolvidas, e após estas pesquisas, necessita-se de uma avaliação em grande escala das propostas mais adequadas a estes requisitos, definidos a partir daí os padrões que garantirão a operação e ajudarão a reduzir a complexidade.

Modelos de negócio: a IoT precisa que novos modelos de negócio sejam criados, assim como melhor explorados os existentes. É importante também a ampliação de objetos e serviços inovadores, que precisam ser desenvolvidos.

Implicações sociais e éticas: a influência da IoT em nossas vidas já é realidade, por isso, não apenas aspectos técnicos e inovadores devem ser tratados, e sim sua influência no comportamento social.

IoT Governança: assim como acontece num ambiente comum de *software*, o governo do ambiente precisa ser feito de forma eficiente. Para que isso funcione na IoT, necessita-se de uma estrutura que o sustente quanto a novos modelos e mecanismos que cubram aspectos legais

e garantam a confiança adequada, o gerenciamento de identidade e as responsabilidades.

Cooperação Internacional: como a IoT possui um âmbito global, ou seja, "coisas"globalmente distribuídas e que necessitam de uma interoperabilidade, se faz necessário o envolvimento e consentimento das partes envolvidas para que essa questão seja garantida.

Integração dos resultados de outras disciplinas: por ser um modelo recente, não possui uma base de consulta sólida, sendo assim, pode ser necessário o estudo de áreas como a robótica, nanotecnologia, biomedicina e ciências cognitivas que colaborem no desenvolvimento e avanço da IoT.

A IoT pode ser vista como um modelo que busca a integração de objetos do dia a dia à Internet, de forma que possam interagir entre si e com as pessoas, produzindo e fornecendo dados e informações a partir desta interação. Dessa forma, a *fog computing* é inserida na IoT (YI; LI; LI, 2015) e (BONOMI et al., 2012), objetivando resolver problemas específicos de algumas aplicações também voltadas para este cenário.

2.3 Fog e Computação em Nuvem

A nuvem apresenta uma nova proposta de utilização de recursos baseada no “pague conforme o uso”, centralizando recursos que podem ser acessados virtualmente e de qualquer lugar através da Internet. A nuvem possui características como disponibilidade e elasticidade, além de fornecer os seguintes serviços em camadas: *Infraestruture as a Service* (IaaS), *Plataform as a Service* (PaaS) e *Software as a Service* (SaaS).

Com o rápido desenvolvimento das aplicações de Internet móvel e IoT, a computação em nuvem convencional, abrange o fornecimento de recursos com centralização global. Apesar de seu uso crescente (YI; LI; LI, 2015), está enfrentando sérios desafios, como alta latência, baixa eficiência espectral, tipo de comunicação de máquina não adaptativa (AI; PENG; ZHANG, 2018), não confiável, falta de suporte à mobilidade e reconhecimento de localização (YI; LI; LI, 2015).

Motivada para resolver esses desafios, a *fog computing* se propõe fornecer recursos e serviços flexíveis da nuvem na borda da rede, a fim de otimizá-los e aumentar o desempenho das redes de comunicação (AI; PENG; ZHANG, 2018; GOMES et al., 2018; CASTILHO G. U.; KAMIENSKI, 2018).

O fato é que muitos aplicativos exigem a localização da *fog* e a globalização da nuvem, principalmente para análise e Big Data (BONOMI et al., 2012). Apesar da principal característica da *fog* ser a de auxiliar e suprir deficiências da nuvem, a computação, armazenamento e recursos de rede são o alicerce de sustentação dessas plataformas (BONOMI et al., 2012).

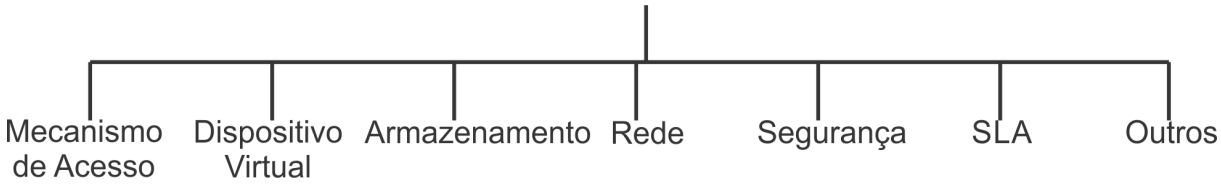
2.4 Gerenciamento de Recursos, QoS e SLA

As características da *fog* citadas na seção 2.1 e expostas por (BONOMI et al., 2012), visam tanto a melhoria de desempenho de aplicativos quanto a eficiência de recursos. Estes, numa arquitetura de *fog* precisam ter sua continuidade estendida da borda à nuvem, exigindo uma estratégia de gerenciamento dos mesmos através do controle adequado (MASIP-BRUIN et al., 2018). O gerenciamento de recursos auxilia no alcance da QoS desejada e no processo de comercialização da *fog*, que precisa ser contabilizada e monitorada nas diferentes camadas (YI; LI; LI, 2015; MAHMUD; KOTAGIRI; BUYYA, 2016).

A QoS é muito importante para o serviço de *fog computing* (YI; LI; LI, 2015) e é necessário que ela seja mantida nos nós da *fog* (MAHMUD; KOTAGIRI; BUYYA, 2016), porém seus atributos nesse ambiente divergem dos tradicionais (LI; LI; ZHAO, 2014), uma vez que a natureza e os tipos de recursos entregues são diferentes (ALHAMAD; DILLON; CHANG, 2010). Dessa forma, Yi, Li e Li (2015) considera que a QoS pode ser dividida em quatro aspectos - conectividade, *delay*, capacidade e confiabilidade - e Mahmud, Kotagiri e Buyya (2016) cita várias métricas para o provisionamento de recursos. Segundo Mouradian et al. (2018), para o fornecimento de um QoS aceitável em ambientes como a *fog*, é essencial a utilização de técnicas de gerenciamento de SLA adequadas. Os autores consideram abordagens de SLA de gerenciamento de nuvem como provável solução para a *fog computing*.

Conforme Alhamad, Dillon e Chang (2010), SLA é um contrato acordado entre o fornecedor e consumidor do serviço, através do qual se pretende definir claramente os termos do serviço fornecido, como o desempenho, disponibilidade e faturamento. É interessante que o mesmo também contemple as obrigações e as ações que serão tomadas em caso de descumprimento por ambas as partes. O SLA é utilizado pelos provedores de serviços para a otimização do uso de sua infraestrutura a fim de garantir a QoS aos consumidores. Teckelmann, Reich e Sulistio (2011) cita o SLA como um dos blocos que podem ser essenciais para a obtenção de interoperabilidade do serviço de IaaS da nuvem, mostrando através de uma taxonomia que pode ser vista na Figura 3 abaixo. Segundo esses autores, o SLA se insere principalmente nas áreas de arquitetura, formato de modelo, gerenciamento e de seus objetivos.

Figura 3 – Taxonomia de Interoperabilidade de IaaS.



Fonte: adaptado de Teckelmann, Reich e Sulistio (2011).

Duas especificações existentes de SLA são consideradas por Alhamad, Dillon e Chang (2010) como principais: contrato de serviço da Web (*WS-Agreement*) e linguagem e estrutura

de acordo de nível de serviço da Web (WSLA). Embora projetadas para descrever a sintaxe do SLA para a nuvem, sua principal deficiência é de não serem dinâmicas. Assim, propõem uma arquitetura básica para desenvolvimento do contrato de SLA, apresentando e definindo os parâmetros utilizados em duas categorias de métricas - de desempenho e de negócio -. Estas são consideradas como principais num ambiente de nuvem, englobando os quatro serviços oferecidos pela mesma (IaaS, PaaS, SaaS e armazenamento como serviço).

3
Trabalhos Relacionados

Neste capítulo, discutimos sobre estudos considerados relevantes e como chegou-se até eles por meio do processo de mapeamento sistemático (PETERSEN et al., 2008) realizado. O objetivo principal é a redução de viés de pesquisa. Sendo assim, foram selecionados estudos sobre *fog computing* para ambientes inteligentes, principalmente de *smart city* ou *smart home* que abordam análise de desempenho, sendo que alguns deles podem direcionar trabalhos futuros.

3.1 Mapeamento Sistemático

O mapeamento sistemático (MS) provê de forma mais abrangente uma estrutura de pesquisa capaz de gerar resultados visuais desses estudos com menor esforço que a revisão sistemática (PETERSEN et al., 2008). Dessa forma, foi seguido o processo de mapeamento detalhado em (SILVA et al., 2018) através dos seguintes passos: definição das **questões de pesquisa** com o intuito de direcionar o mapeamento; realização da pesquisa através da **estratégia de busca e de seleção** dos estudos primários para a obtenção de melhores resultados; **critérios de seleção** que facilitam a triagem dos artigos relevantes e a **análise dos resultados**, através da qual os dados que respondem às questões de pesquisa são extraídos.

Para um melhor entendimento e organização deste artigo, cada passo foi subdividido em duas subseções: uma definida **estado da arte**, representa os artigos científicos. A outra como **estado da técnica**, referencia as patentes.

3.1.1 Questões de Pesquisa

Como auxílio no alcance do objetivo deste estudo, foram levantadas duas questões para o estado da arte e mais uma para o estado da técnica, cujas respostas foram obtidas através do mapeamento dos respectivos dados retirados das bases de artigos e de patentes. Segundo Petersen

et al. (2008), as questões de pesquisa são o primeiro passo do mapeamento sistemático, e todos os demais devem estar direcionados a ele.

3.1.1.1 Estado da Arte

Esta subseção aborda as questões relacionadas ao mapeamento do estado da arte (QEA1 e QEA2), bem como os dados que serão mapeados dos artigos para a obtenção das respostas às mesmas.

QEA1 - Qual a conjuntura da *fog computing* para aplicações de ambientes inteligentes com análise de desempenho?

A resposta a essa questão irá proporcionar um *overview* sobre a *fog computing* aplicada a sistemas voltados para esses ambientes inteligentes, culminando na detecção dos seguintes dados: quantidade de artigos existentes; publicações por ano, autores e fontes que contribuem com a área.

QEA2 - De que maneira a composição das arquiteturas de *fog* utilizadas nos artigos contribuem para os ambientes inteligentes e como são medidas?

Essa questão visa abordar como a estrutura da *fog computing* vem sendo sugerida pela comunidade científica para atender às necessidades de aplicações para ambientes inteligentes. Identifica as plataformas, dispositivos, ferramentas, métricas, ambientes, recursos atendidos, bem como a análise dos ambientes utilizados nos experimentos.

3.1.1.2 Estado da Técnica

Nesta subseção é apresentada uma questão de pesquisa relacionada ao mapeamento do estado da técnica (QET1) além dos dados que serão extraídos para respondê-la.

QET1 - Qual o atual status das patentes relacionadas a *fog computing* para ambientes inteligentes?

Essa questão visa fornecer uma visão geral sobre o estado das patentes. Como a contribuição das bases utilizadas na pesquisa base; a quantidade de patentes por ano de publicação; os países; inventores e requerentes que mais atuam na área; a classificação internacional das patentes (CIP); o ambiente para o qual a invenção foi planejada; o invento patenteado e como ele utiliza a *fog computing*.

3.1.2 Estratégia de Busca e de Seleção

Para dar início às pesquisas, foram definidas características que contribuíssem para a obtenção de uma maior abrangência dos resultados de pesquisa relacionados à área pesquisada.

Para reduzir a possibilidade de vieses, foram selecionadas bases que dispusessem das seguintes características gerais: consultas através da Internet; presença de mecanismos de busca

através de palavras-chaves; que contemplassem artigos ou patentes na área de computação; que disponibilizassem de conteúdos atualizados e que fossem valorizadas pela comunidade científica.

3.1.2.1 Estado da Arte

Para a pesquisa dos estudos primários, que segundo Petersen et al. (2008) são experimentos controlados, surveys e estudos de caso representados em formatos de artigos científicos, foram utilizadas quatro bases de busca que possuem as características definidas acima. A *Scopus* (SCOPUS, 2018), que compreende outras bases, como a *Elsevier*, *Springer*, além das também selecionadas *IEEE* (IEEE, 2018) e *ACM* (ACM, 2018), incluídas na seleção para reduzir as chances de perder informações importantes. Somadas com a *Sciencedirect* (SCIENCEDIRECT, 2018), elas abarcam periódicos cientificamente relevantes para a computação.

O acesso a essas bases se deu por meio do *site* de periódicos da CAPES (CAPES, 2018) através do usuário cedido pela Universidade Federal de Sergipe (UFS). Assim, foi possível o *download* completo das publicações encontradas.

A identificação dos estudos primários foi feita através da utilização de *string* de busca nas bases selecionadas. Uma *string* de busca é composta por palavras-chave e operadores lógicos, como os "AND" e "OR" que compuseram a *string* utilizada neste MS. Já as palavras-chave mais comuns neste nicho, foram identificadas através de pesquisa prévia, onde a princípio foram selecionadas quatro (*performance analysis, smart home, smart city e fog computing*). Porém, os resultados não foram satisfatórios, encontrando apenas 4 artigos no total. Sendo assim, foram analisados alguns estudos relacionados ao tema, e observadas palavras distintas das anteriores, mas que abordavam o mesmo assunto, como por exemplo a *testbed* que possuía conteúdo similar aos trabalhos que referenciavam a *performance analysis*. Somado a isso, a inclusão de sinônimo e plural, trouxe melhores resultados através das palavras *smart living* e *smart cities*, refletindo *smart home* e *smart city* respectivamente.

As palavras *cloud computing* e *Internet of Things* não foram citadas, uma vez que elas são intrínsecas à *fog computing*, *smart city* e *smart home*, resultando na seguinte *string* de busca:

(fog AND ("smart living"OR "smart home"OR "smart city"OR "smart cities") AND (testbed OR "performance analysis"))

Esta *string* foi executada no dia 04 de outubro de 2018 e adaptada a cada fonte de busca, condicionando a pesquisa à presença dos termos nos títulos, resumos ou palavras-chave.

Considerando os trabalhos na língua portuguesa, foi montada uma outra *string* de busca com as seguintes palavras: *fog*, névoa, neblina, casa inteligente, cidade inteligente e análise de desempenho. Como não retornou nenhum resultado, foram incorporadas a uma *string* única em nova tentativa, o que também não surtiu qualquer efeito. Destarte, foram removidas com o propósito de se manter uma composição mais legível e enxuta.

Com a *string* definida, foram executadas as buscas em cada base da seguinte forma: na

base *Sciencedirect*, foi escolhida a opção de busca avançada e informada a *string* no campo *"Title, abstract or keywords"*, retornando apenas 1 resultado. Na do *IEEE*, também foi efetuada uma busca avançada através da guia *"Command Search"* e da opção *"Metadata Only"*, obtendo 11 trabalhos encontrados. Já na *Scopus*, a *string* foi executada na guia *"Documents"* resultando em 16 estudos. A *string* digitada no campo *"Edit Query"* da fonte *ACM*, apesar de ajustada conforme requisitos dessa base, não retornou nenhum resultado. Para uma melhor visualização dos mesmos, a tabela 1 exibe os 28 estudos primários encontrados, sendo que 57% deles foram localizados na base da *Scopus*. Devido à ausência de resultados na fonte *ACM Digital*, a tabela não a referencia.

Tabela 1 – Quantitativo de Resultados dos Estudos Primários por Base.

Bases de Busca	**Estudos Primários**
Sciencedirect	1
IEEE	11
Scopus	16
TOTAL	**28**

3.1.2.2 Estado da Técnica

Além de possuírem as características definidas na seção 3.1.2, a *World Intellectual Property Organization* (WIPO) (WIPO, 2018) e o Instituto Nacional da Propriedade Industrial (INPI) (INPI, 2018) foram as bases de depósitos de patentes selecionadas por possuírem uma abrangência mundial e nacional (Brasil) respectivamente. Focada no desenvolvimento de um sistema internacional de propriedade intelectual, a WIPO é vinculada a Organização das Nações Unidas (ONU) e contabiliza 191 países membros. Já o INPI é uma autarquia federal Brasileira responsável por questões de propriedade intelectual para a indústria. O acesso a essas bases se dá de forma gratuita, atendendo a uma das características da publicação de patentes.

Uma *string* de busca foi formada para cada base escolhida, e para ampliar o escopo da busca, no *site* do INPI foram utilizadas apenas as palavras-chave *fog computing* e também em português, computação em névoa, mesmo assim não foi localizada nenhuma patente. Já para a base *WIPO*, foram selecionadas as palavras *fog computing, smart living, smart home, smart city* e *smart cities*. A *string* foi formada de maneira a identificar essas palavras no título, resumo e no campo reinvindicações do documento de patente, ficando da seguinte forma:

FP:(EN_TI:((fog computing ("smart living"OR "smart home"OR "smart city"OR "smart cities"))) OR EN_AB:((fog computing ("smart living"OR "smart home"OR "smart city"OR "smart cities"))) OR EN_CL:((fog computing ("smart living"OR "smart home"OR "smart city"OR "smart cities"))))

A *string* usada na base *WIPO* foi executada no dia 29 de outubro de 2018 através da pesquisa avançada, e obteve como resultado 6 patentes. Como não foi encontrado nenhum resultado na base do INPI, este artigo totalizou apenas as 6 patentes encontradas na base *WIPO*.

3.1.3 Critérios de seleção

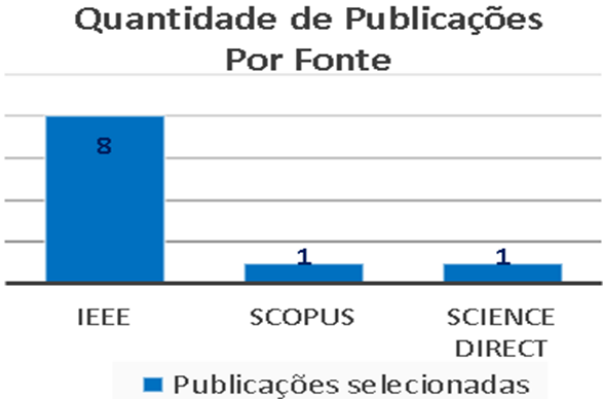
Nesta seção foram definidos critérios de inclusão e exclusão, pois segundo Petersen et al. (2008) eles são importantes para se obter uma maior precisão na seleção dos trabalhos relacionados às questões de pesquisa.

3.1.3.1 Estado da Arte

Para a triagem dos estudos relevantes, foi estabelecido primeiramente como critério de inclusão, que os artigos encontrados apresentassem análise de desempenho ou experimento da *fog computing* em ambientes inteligentes e contemplassem modelos de arquitetura, técnicas ou métodos aplicados à mesma. Como critérios de exclusão, os aspectos definidos foram os estudos duplicados e artigos que abordem outras plataformas diferentes da *fog computing*. Vale ressaltar que basta o artigo se enquadrar em apenas um dos critérios de exclusão para ser eliminado, e para ser selecionado, deve atender a todos os requisitos de inclusão.

A partir dos 28 estudos primários encontrados, 10 foram selecionados após a aplicação dos critérios de inclusão e exclusão. A figura 4 mostra a quantidade de publicações por fonte, após a aplicação desses critérios.

Figura 4 – Quantidade de Publicações Selecionadas por Fontes.



Fonte: Autoria Própria.

A tabela 2 exibe um quantitativo dos estudos primários encontrados e dos que foram considerados relevantes com base nos critérios de seleção.

Tabela 2 – Quantitativo de Resultados das Buscas e Seleção.

Bases de Busca	**Estudos Primários**	**Estudos Relevantes**
Sciencedirect	1	1
IEEE	11	8
Scopus	16	1
TOTAL	**28**	**10**

Apesar de ter sido a fonte com o maior número de trabalhos encontrados, a *Scopus* detectou 9 artigos também encontrados na *IEEE*, sendo esses eliminados pelo critério de exclusão "artigos duplicados". Os outros 9 arquivos excluídos, não se enquadraram no critério de inclusão que obriga a contemplação de conteúdo direcionado à *fog computing* para ambientes inteligentes. Desta forma, o processo de filtragem dos resultados finalizou com um total de 10 trabalhos selecionados.

3.1.3.2 Estado da Técnica

Foi definido como critério de inclusão da patente, que abordassem algum sistema, tecnologia ou técnica para ambientes inteligentes utilizando a *fog computing*, ou para a *fog computing* direcionada para aplicações de *smart city* ou *smart home*. As patentes duplicadas ou publicadas em outra língua que não fosse a portuguesa ou inglesa, entraram no critério de exclusão. Após a aplicação desses critérios, das 6 patentes encontradas apenas 1 foi considerada relevante para este mapeamento. As outras 5 patentes, mesmo possuindo números de aplicação, publicação e concessão diferentes, foram removidas pelo critério de exclusão de patentes duplicadas, pois se tratavam da mesma tecnologia.

3.1.4 Análise dos Estudos Relevantes

A *fog computing* foi proposta como uma arquitetura situada na borda da rede que lida diretamente com a nuvem e com os dispositivos finais para suprir as necessidades das mais variadas aplicações e dispositivos. Diante disso, eles vêm propondo soluções para os problemas enfrentados pela mesma. Nesta seção, assim como nas anteriores, os resultados do estado da arte e do estado da técnica serão analisados separadamente através das seguintes subseções.

3.1.4.1 Estado da Arte

Masip-Bruin et al. (2016), apresentam uma arquitetura denominada *fog-to-cloud* (F2C) em camadas e compara com uma F2C otimizada (OF2C) e a nuvem tradicional, apresentando através de simulação e caso de uso em assistência médica os benefícios da execução de serviços paralelos nas diferentes camadas F2C. Como resultado, obtidos com o uso das ferramentas Tareador e Paraver, os autores demonstram uma melhoria na velocidade de execução da tarefa de 32,05% da arquitetura OF2C em relação à nuvem tradicional, e coloca como desafio a criação de estratégias de gerenciamento dos recursos nas diferentes camadas da F2C para fornecer QoS.

Bruneo et al. (2016) implementam através de simulação um *framework* denominado Stack4Things, estrutura baseada no OpenStack que abrange IaaS (*Infrastructure as a Service*) e PaaS (*Plataform as a Service*). Também apresentam um estudo de caso de coleta de dados ambientais por meio do #SmartME (projeto para estimular a criação de um novo ecossistema virtual de *smart city* para a cidade de Messina). Este artigo nos dá mais uma indicação dos tipos de serviços que será fornecido pela *fog*, reforçando que a sugestão levantada por Mouradian et

al. (2018), de que a adaptação do SLA da *cloud* para a *fog computing* possa ser uma possível solução para a implementação desse acordo e também como auxílio à QoS.

Hong, Tsai e Hsu (2016) desenvolvem uma arquitetura de *fog* utilizando ferramentas de virtualização (Docker Container e Kubernets), que envia aplicações para os dispositivos IoT. Implementam e avaliam 2 algoritmos, o MDA, que fornece soluções quase ótimas, resolvendo 1000 solicitações com 500 dispositivos em menos de 2 segundos, enquanto o OPT leva 80 segundos para resolver 20 pedidos com 4 dispositivos. A *fog* consegue implantar 20 solicitações em 9 segundos.

Santos et al. (2017) abordam a alocação de recurso na fog através do algoritmo ILP e fazem uma análise baseada em casos de uso no âmbito da *city of things* de Antuérpia. Identificam que para 200 solicitações de serviço, cada configuração de modelo requer em média 23 minutos para encontrar a solução ideal. Além disso, a contagem média de saltos diminui enquanto os pedidos aumentam, mesmo se na configuração do modelo não objetivar a otimização relacionada à latência.

Amadeo et al. (2017) apresentam uma plataforma *cloud of things* (COT) de três camadas com virtualização denominada ICN-iSapiens, para serviços de *smart home*. Utilizam como ferramentas o CCN-Lite e VOContainer com dispositivos genéricos (Raspberry‘s).

Carmo et al. (2017) proveem o compartilhamento eficiente de recursos de rede do cliente através da criação de camadas de rede configuradas utilizando SDN e VNF implementados em dispositivos comuns e de baixo custo ao usuário (Ex: Raspberry), objetivando trazer serviços personalizados e sem fio mais perto dos dispositivos móveis e sensores. Como resultado, o atraso médio da nuvem foi de aproximadamente 133ms, contra 12 e 5,3ms para computador de placa única e PC respectivamente.

Santos et al. (2018) apresentam uma solução de detecção de anomalias para aplicação de *Smart City* baseada em *fog* conectada por LPWAN e avalia através de algoritmos no testbed da cidade de Antuérpia. Os resultados mostram que tanto o agrupamento *Birch* quanto os mecanismos de detecção de anomalias RC podem ser executados por recursos de *fog*. As tecnologias LPWAN avaliadas e validadas para a aplicação de qualidade do ar foram: IEEE 802.11ah, DASH7 e LTE-M.

Masip-Bruin et al. (2018) estudam a questão da continuidade de recursos e do gerenciamento coordenado da *fog* e da nuvem, e propõe os blocos fundamentais para a arquitetura do sistema. Demonstram os benefícios de uma abordagem de gerenciamento em camadas ao considerar o tamanho e o tempo de pesquisa dos bancos de dados da *smart city*. Os autores verificam que quanto menor a área da cidade, menor o tamanho do banco de dados e o tempo de pesquisa, no entanto, quanto menores as áreas, menor o número de serviços a serem executados, portanto, menor o interesse nesses serviços pelos usuários.

Neto et al. (2018) desenvolvem uma estrutura FISVER baseada em *fog* para melho-

ramento de Vigilância em *Smart Transportation Safety* (STS) através da detecção de objetos. Indicam uma notável melhoria de desempenho na CPU (27,76%), na rede (51,98%) e na economia de energia (62. 14%) em comparação com uma implantação típica.

Amadeo et al. (2019) apresentam soluções que dependem do ICN para monitorar e controlar o ambiente inteligente e discutem a integração do ICN com os recursos da *fog computing*, apresentando uma arquitetura de referência junto com um *testbed* preliminar.

De acordo com esses estudos, identificamos uma maior inclinação das pesquisas para ambientes de *smart city*, utilizando como principais métodos de análise o *testbed*, simulação e provas conceituais, respectivamente. Quanto às ferramentas, observa-se a utilização de uma variedade delas com diferentes objetivos associados à cada artigo. Porém é notória a priorização no uso de soluções virtuais, como máquinas virtuais e contêineres.

Há também uma forte tendência pela escolha de dispositivos genéricos (Ex: Raspberry e Galileo) e minicomputadores em ambientes de IoT, que pode ser comprovada pelo uso mais frequente desses dispositivos nos trabalhos selecionados. Além deles, os smartphones e sensores também são utilizados como dispositivos finais (*endpoints*).

3.1.4.2 Estado da Técnica

A quantidade de patentes encontradas, revela a oportunidade de investimento nesta área. Para facilitar a análise desse resultado, bem como as respostas à questão QET1, foi inserido através da figura 5 o resultado integral da busca realizada na base *WIPO*.

Antes de analisar essa figura, vale lembrar que apesar de conter 6 patentes, todas se referem à mesma tecnologia e por isso foram consideradas como apenas 1 estudo relevante.

Figura 5 – Resultado da Pesquisa de Patentes Realizada na Base WIPO.

Countries		IPC		Inventor		Applicant		Pub Date	
Name	No	Name	No	Name	No	Name	No	Date	No
United States	4	A61H	6	Christopher Gary Henshue	4	Brandbumps, LLC	5	2017	3
Australia	1	E01C	4	Gary LaVerne Henshue	4	BRANDBUMPS, LLC	1	2018	3
PCT	1	G08B	4	James Cyrus Rice	4	Brandsbumps, LLC	1		

Fonte: Wipo (2018).

A figura 5 mostra uma tabela contendo 5 colunas relacionadas às seguintes informações que serão analisadas:

Countries - Refere-se aos países nos quais as patentes foram publicadas, revelando quais deles tem contribuído nesta área, nesse caso, os Estados Unidos lideram com 4 publicações. Vale observar que entre eles é exibido o acrônimo PCT, que em português significa Tratado de

Cooperação de Patentes. Segundo o Wipo (2018), é um tratado que auxilia os países envolvidos em relação à propriedade intelectual e facilita o pedido internacional de patentes.

IPC - *International Patent Classification*, ou CIP em português, classifica os pedidos de patente de acordo com a área (SILVA et al., 2018). Algumas patentes utilizam mais de um código CIP, então todos eles foram contabilizados, sendo que as 6 patentes foram classificadas na área de "aparelho de terapia física, por exemplo. dispositivos para localizar ou estimular pontos reflexos no corpo; respiração artificial; massagem; dispositivos de banho para aparelhos terapêuticos ou higiênicos especiais ou partes específicas do corpo" (A61H), 4 como "construção de, ou superfícies para estradas, terrenos esportivos, ou como máquinas ou ferramentas auxiliares para construção ou reparo" (E01C) e 4 como "sistemas de sinalização ou chamada; ordem de telegrafias; sistemas de alarme" (G08B).

Inventor – São os criadores das patentes. Dado que informa os principais contribuintes da área. Nesse caso, 3 inventores estão informados em 4 patentes cada.

Applicant – É o requerente da patente. Nesse caso, apesar de aparecerem 3 requerentes para as 6 patentes, a similaridade entre eles sugere que se trata da mesma empresa.

Pub Date – Refere-se a data de publicação das patentes, que na ocasião foram publicadas 3 em 2017 e 3 em 2018.

O invento de Henshue, Henshue e Rice (2017) se trata de um aparelho de painel de aviso tátil e sistema com tecnologia inteligente que apresenta uma variedade de funcionalidades, como questões de segurança e acessibilidade numa *smart city*. Essa tecnologia multiuso foi projetada de forma a suportar sua integração com outras tecnologias de conectividade e arquiteturas como a *fog computing*, podendo funcionar como uma matriz de processadores distribuídos.

3.2 Comparação dos Estudos Relevantes

Além da análise resumida na seção anterior de cada estudo considerado relevante para o nosso tema, apresentamos através da tabela 3 um comparativo entre os mesmos e este trabalho através de algumas características. Para esta comparação consideramos apenas os trabalhos do estado da arte, uma vez que a patente encontrada, por focar no produto em si, não contempla detalhes sobre uma arquitetura de *fog*.

Tabela 3 – Comparação Entre os Trabalhos Relacionados.

Referência	QoS	SLA	Gerenciamento de Recursos	Heterogeneidade	Interoperabilidade
Masip-Bruin et al. (2016)			X		
Bruneo et al. (2016)			X		
Hong et al. (2016)					
Santos et al. (2017)					
Amadeo et al. (2017)			X	X	
Carmo et al. (2017)					
Santos et al. (2018)			X		
Masip-Bruin et al. (2018)			X		
Neto et al. (2018)					
Amadeo et al. (2019)			X		
Este Trabalho	**X**	**X**	**X**	**X**	**X**

A maioria dos artigos avaliam seus ambientes através de algumas métricas definidas, sendo que as mais utilizadas por esses artigos, foram processamento (MASIP-BRUIN et al., 2016; HONG; TSAI; HSU, 2016; CARMO et al., 2017; NETO et al., 2018) e largura de banda (HONG; TSAI; HSU, 2016; CARMO et al., 2017; NETO et al., 2018), sendo pouco exploradas as métricas para a análise da rede, como por exemplo a largura de banda de *uplink* e *downlink* e o custo de uso do lado do usuário. Nenhum deles também se aprofundam em questões relacionadas à QoS e SLA, apenas as indicando para trabalhos futuros conforme direcionado por Masip-Bruin et al. (2016) e Masip-Bruin et al. (2018).

A característica de gerenciamento de recursos é compreendida pela maioria dos trabalhos, uma vez que fazem o monitoramento dos recursos disponibilizados nos ambientes propostos a fim de validar ou aferir as métricas impostas. Esta é uma característica importante para a continuidade dos serviços, bem como a garantia da QoS.

Por fim, apenas Amadeo et al. (2017) discute a questão da heterogeneidade, fornecendo na camada da *fog* um servidor que hospeda uma aplicação de gerenciamento e oculta detalhes dos *endpoints* através de representação virtual da rede.

Conforme apresentado na tabela 3, este trabalho preenche algumas lacunas existentes nesta área, bem como resolve alguns problemas conforme explicado no capítulo 4.

4

Arquitetura de Fog Computing Virtualizada

Este capítulo descreve as características e funcionalidades da arquitetura proposta, a qual chamaremos a partir deste momento de arquitetura de *fog computing* virtualizada (AFCV), que é a principal contribuição desta dissertação de mestrado. Aspectos relacionados ao gerenciamento de recursos, QoS e SLA, além do que é fornecido a nível de serviço e como tudo isso pode auxiliar provedores e clientes também são discutidos no mesmo.

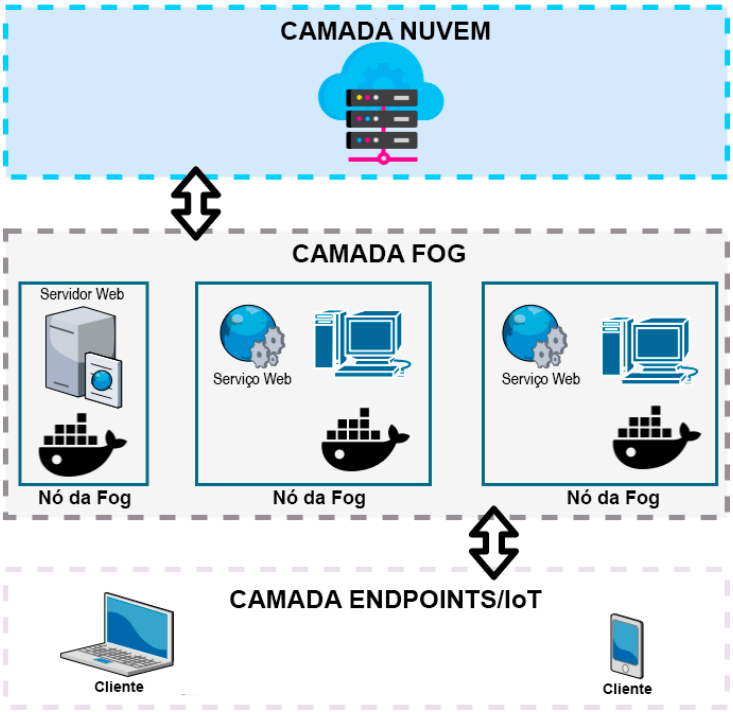
4.1 Camadas da AFCV

A implementação da AFCV para fornecimento de gerenciamento de recursos, QoS e SLA, foi verificada por meio da análise de desempenho em ambiente real, e teve como base o ambiente proposto por (MAHMUD; KOTAGIRI; BUYYA, 2016), preservando um menor número de camadas, sem descaracterizar os conceitos e estrutura da *fog*.

A AFCV é desmembrada em três camadas, suportando diferentes dimensões como o nível de aplicação e de rede, que funcionam de forma independente e personalizada. Todos os níveis foram gerenciados com base nas métricas estabelecidas de acordo com os trabalhos realizados por (MOURADIAN et al., 2018; LI; LI; ZHAO, 2014; JAIN, 1991) e pelos estudos relevantes citados na subseção 3.1.4, dentre elas, a largura de banda de *uplink* e *downlink* que foi medida até o nível de usuário a fim de validar o SLA predefinido e estabelecer a qualidade de serviço.

A figura 6 apresenta esses diferentes níveis da AFCV, através dos quais se buscou um ambiente simplificado, de alto nível e capaz de atender a diferentes demandas.

Figura 6 – Camadas da AFCV.



Fonte: Própria do autor.

Suas camadas são descritas a seguir:

Camada Nuvem: Representa a camada superior da AFCV que interage com a *fog*, fornecendo à mesma recursos como autenticação, acessos externos, dentre outros em que a *fog* seja incapaz de suprir.

Camada *Fog*: Camada intermediária situada na borda da rede, próxima aos *endpoints* (dispositivos finais/clientes), cuja finalidade é reduzir a latência de serviços e garantir uma melhor qualidade no fornecimento dos mesmos (QoS), tornando possível o atendimento aos requisitos negociados com os clientes e assim garantir o cumprimento de contratos de SLA's através do planejamento, configuração e gerenciamento. Esta camada também é responsável por se comunicar com a nuvem, repassando requisições que ela seja incapaz de atender, seja a nível de recursos, segurança ou aplicação.

Camada *Endpoints/IoT:* Esta é a camada mais baixa da AFCV, constituída por dispositivos de diversos tipos, sejam eles voltados para a IoT ou utilizados por usuários finais, como *notebooks*, celulares e *desktops*. Tais dispositivos são capazes tanto de consumir quanto fornecer informações para a camada *fog*. Sendo assim é importante que as medições de rede sejam estabelecidas incluindo esta camada, percorrendo toda a rota do início ao fim (MOURADIAN et al., 2018).

Como a AFCV é composta por objetos providos de recursos finitos (CPU, memória, disco e largura de banda), a divisão da mesma em camadas favorece a distribuição dos serviços

fornecidos por uma aplicação, o escalonamento horizontal, o gerenciamento e melhor aproveitamento dos recursos, prezando por uma infraestrutura acessível e adaptável, capaz de garantir a interoperabilidade entre provedores de serviço.

Para suportar a heterogeneidade e dinamismo dos ambientes de IoT, as funções de rede virtual (VNF) foram implementadas nos nós da *fog* através de contêineres criados com a ferramenta Docker Container. Os contêineres são componentes de virtualização que requerem menos recursos e são capazes de serem instalados em curto tempo (HONG; TSAI; HSU, 2016). Esses autores afirmam que os módulos virtualizados facilitam sua colocação dinâmica nos dispositivos ou migração para otimização. Essa ferramenta também controlará o ciclo de vida das VNF's (criação, gerenciamento e finalização) (CARMO et al., 2017).

Maiores detalhes sobre sua capacidade, prototipação e avaliação são apresentados a partir da próxima seção 4.2. Nela são descritas as ferramentas que a constituem, bem como a metodologia e sua aplicação.

4.2 Prototipação e Avaliação

4.2.1 Metodologia

Segundo Wazlawick (2009), é fundamental que se defina o método de pesquisa que aponta os caminhos para se atingir os objetivos descritos. Como guia para estes caminhos, é interessante a utilização de uma abordagem sistemática como a proposta por Jain (1991), cujo método será utilizado nesta prototipação. Dessa forma, o primeiro passo deve ser a definição dos objetivos da prototipação e da avaliação, uma vez que as demais etapas são direcionadas para o atingimento dos mesmos.

O segundo passo consiste na definição do sistema e serviços providos pela AFCV, já que existem conjuntos de possíveis resultados para cada um deles. O terceiro é a seleção da técnica de avaliação, bem como a divisão das etapas onde os passos seguintes estão inseridos, objetivando melhor estruturar a implementação do ambiente. O quarto corresponde a especificação das métricas mais apropriadas que se relacionam com os atributos de QoS e SLA na AFCV (LI; LI; ZHAO, 2014), evitando métricas redundantes e com alta variação (JAIN, 1991).

Já no quinto passo temos a escolha dos fatores que sofrerão variações durante a pesquisa, para que possam ser estudados. Os fatores e níveis de variação poderão ser aumentados conforme a necessidade. O sexto corresponde a escolha e instalação da aplicação que irá gerar a carga no sistema e para os *softwares* de medição. No sétimo ocorre a implementação da AFCV em ambiente real e a configuração da mesma, atendendo aos requisitos dos passos anteriores.

O oitavo passo é o do planejamento e execução dos experimentos com base na lista de fatores para a geração da carga, e do estabelecimento de uma sequência para prover o máximo de informações possíveis. O nono corresponde a análise e interpretação dessas informações através

de *softwares* e técnicas estatísticas adequadas para a obtenção de conclusões consolidadas.

Já o décimo é o da apresentação dos resultados. Ele contempla a sumarização dos resultados, bem como seu relacionamento e variação com os pré-requisitos de QoS. Também farão parte dos resultados, a identificação das limitações e a sobrecarga ou subutilização do sistema distribuído, caso existam.

4.2.1.1 Aplicação da Metodologia

Objetivos:

• Implementação da AFCV, alocação e gerenciamento de recursos para provimento de QoS e auxílio aos contratos de SLA;

• Análise de desempenho dos elementos da *fog* bem como do serviço fim a fim;

• Determinar fatores relevantes no que se refere ao desempenho da AFCV.

Sistema e Serviço:

• O sistema inserido na AFCV corresponde a um *software media center* multiplataforma denominado Plex, que funciona no modo cliente servidor e é capaz de armazenar diversos tipos de arquivos de mídia para fornecer entretenimento através de ambientes como por exemplo o de *smart home*.

• Além do armazenamento de mídias, o Plex fornece serviço de *streaming* de vídeo, tornando possível o acesso remoto a todo seu conteúdo.

Técnica de avaliação:

• Medição, pois trata-se de uma técnica útil para análise de desempenho de sistemas reais ou similares.

Para implementação do ambiente real, as atividades foram divididas em cinco etapas:

• Etapa 1 - Projeto de prototipação e cenário de testes;

• Etapa 2 - Especificação das métricas;

• Etapa 3 - Definição da carga de trabalho;

• Etapa 4 - Planejamento e realização dos experimentos;

• Etapa 5 - Análise estatística dos resultados obtidos.

As três primeiras etapas serão detalhadas individualmente nas próximas subseções, já as etapas 4 e 5 serão abordadas juntamente no capítulo 5.

4.2.2 Prototipação e Cenário de Testes

Para a realização da prototipação da AFCV foi necessário passar por todas as etapas anteriores desta dissertação de mestrado, principalmente pelo processo de mapeamento sistemático, através do qual foram extraídas informações sobre as características, melhores práticas e ferramentas utilizadas em um ambiente de *fog*, como nos trabahos de Hong, Tsai e Hsu (2016) e Carmo et al. (2017).

Após este empenho, foi desenhada a AFCV e suas características conforme a seção 4.1. Para a prototipação da mesma em ambiente real foram analisados primeiramente os recursos de *hardware* e infraestrutura disponíveis na Universidade Federal de Sergipe (UFS), onde foi implementada. Dessa forma, para compor as 3 camadas arquiteturais contamos com os dispositivos apresentados na tabela 4.

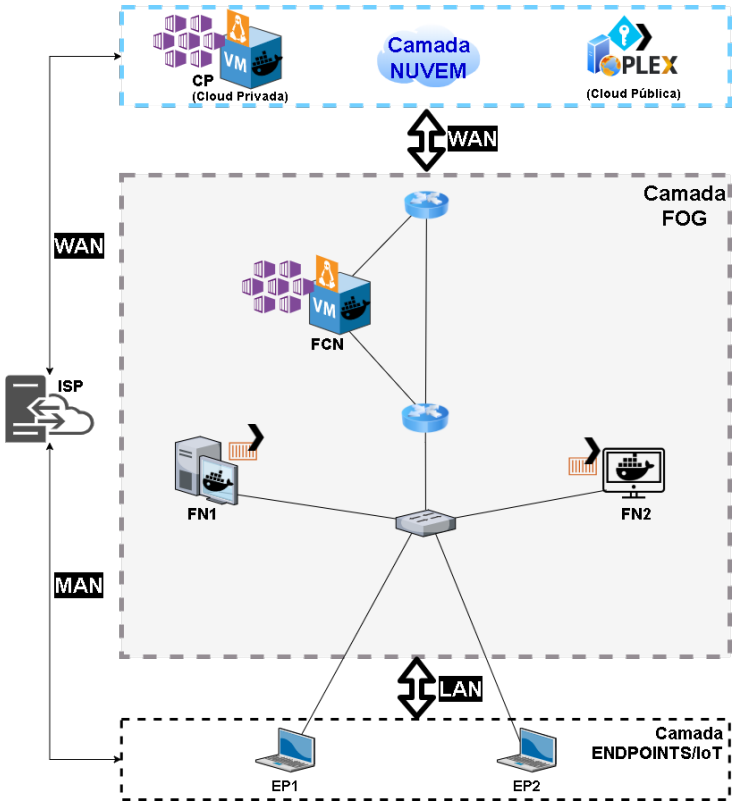
Tabela 4 – Recursos de *Hardware*.

ID	TIPO	CPU TOTAL	MEMÓRIA	DISCO	REDE
CP/FCN	VM	24	100 GB	400 GB	2 of 1000 Mbps
FN1	Desktop	4	8 GB	150 GB	1000 Mbps
FN2	All in One	4	4 GB	500 GB	1000 Mbps
EP1	Laptop	4	8 GB	1 TB	1000 Mbps
EP2	Laptop	4	8 GB	1 TB	1000 Mbps

Diante destes dispositivos, foram instituídas as funções de cada um deles, com base e proporcionalmente às suas respectivas capacidades computacionais, organizando seus posicionamentos nas camadas para a garantia de atendimento à todas as características conceituais da AFCV que foram discriminadas nas seções anteriores.

Vale destacar na tabela 4 a máquina virtual (VM) utilizada. Ela possui 2 ID's e 2 placas de rede por ter dupla funcionalidade na AFCV, onde atua como *nuvem* privada (CP) através de uma placa de rede com um IP de externo configurado e exposto à Internet, e também atua como *Fog Control Node* (FCN) através da outra placa de rede configurada com um IP interno da *fog*. Na figura 7 apresentamos o cenário de prototipação da AFCV, onde essa VM é representada como sendo 2 máquinas distintas para facilitar o entendimento no decorrer deste trabalho, já que sua funcionalidade serve para as duas camadas (nuvem e *fog*).

Figura 7 – Cenário da Prototipação.



Fonte: Própria do autor.

Na camada da nuvem foram utilizados 2 servidores, 1 servidor de nuvem pública de domínio dos mantenedores do *software* central de mídia (Plex), ao qual não é liberado acesso administrativo e portanto, não se pôde dimensionar seus recursos de *hardware* e rede. Na AFCV, este servidor é responsável pela autenticação de usuários no Plex.

O outro servidor (CP) é o ponto chave da AFCV, localizado de forma estratégica no *core* da rede, ou seja, na borda da *fog* com a Internet, ele funciona como uma nuvem privada, atendendo as demandas externas à AFCV, sendo capaz de entregar os mesmos serviços disponibilizados na *fog*. Devido a sua importância e à maior quantidade de funções que esse dispositivo irá desempenhar, foi alocada para esta função a máquina com maior recurso, uma VM com 24 CPU's, 100 GB de memória, 400 GB de disco e 2 placas de rede que separam o tráfego da nuvem e da *fog*.

Já na camada *fog*, operam 3 servidores na sua composição, que representam dois tipos de nós denominados *Fog Control Node* (FCN) e *Fog Node* (FN). Como o primeiro recebe o maior número de responsabilidades, lhe foi atribuída à VM descrita no parágrafo anterior. O segundo possui uma atuação mais limitada, sendo monitorado e gerenciado pelo FCN. Para este fim foram utilizados os dispositivos com menos recursos e com diferentes configurações, o que possibilitará um maior número de conclusões para os resultados da análise.

Por ser o nó de controle e gerenciamento, o FCN recebe a nomenclatura de *manager*

pelo aplicativo Docker Container. Ele é usado para montar o *cluster* de servidores da *fog* e implementar serviços através da conteinerização, que será melhor detalhado na subseção 4.2.2.1. Também foram implementados apenas neste servidor o gerenciador de recursos e o balanceador de carga. Dessa maneira, foi necessária a instalação de um sistema operacional (S.O.) Linux Debian 4.9 e da plataforma Docker Container (versão 19.03.5), através da qual os serviços foram virtualizados na estrutura de contêiner.

Os FN1 e FN2 fornecem os serviços utilizados pelos usuários finais conectados à *fog*, atuando como servidor de mídia (Ex: *streaming* de vídeo e armazenamento) através da ferramenta Plex. Esses dispositivos estão na mesma rede do FCN com quem se comunicam na camada *fog* e consequentemente com os *endpoints* na última camada. Para o fornecimento desses serviços, o FN1 foi alocado num *desktop* contendo um total de 4 CPU's, 8 GB de memória, 150 GB de espaço em disco e configurado com o S.O. CentOS. Já o FN2 é um dispositivo *all in one* com 4 CPU's, 4 GB de memória, 500 GB de disco e S.O. Linux Mint. Nos dois foi instalada a plataforma Docker Container (versão 19.03.6) e o contêiner com o servidor Plex (versão 4.12.3), trabalhando em *cluster* com o FCN através da ferramenta *Docker Swarm*.

A última camada é a *endpoints/IoT*, onde os usuários e dispositivos de IoT estão localizados. Portanto ela é capaz de consumir e/ou fornecer informação. Foram utilizados dois *laptops* (EP1 e EP2) de mesma configuração (4 CPU's, 8 GB de memória RAM, 1 TB de disco e S.O. Windows 10) para consumo dos serviços oferecidos pela *fog*; gerar tráfego; acessar a central de mídia e efetuar as devidas medições do ambiente.

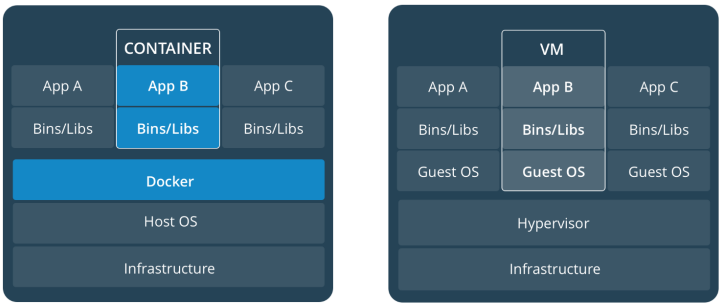
Também podemos identificar através da figura 7 que a composição segue a estrutura de camadas definida na seção 4.1, onde a nuvem continua separada na parte superior e acessada através de uma rede de longa distância (WAN). Em seguida temos as camadas *fog* e *endpoints/IoT* que estão na mesma rede local (LAN).

O provedor de internet (ISP) foi inserido na figura apenas para ilustrar como foi realizado o acesso dos clientes EP1 e EP2 com a nuvem para a realização dos testes referentes à essa camada, mas não faz parte da AFCV.

4.2.2.1 O Docker Container na AFCV

O *Docker Container* (DOCKER, 2018) é uma tecnologia de virtualização diferente das tradicionais, pois possibilita a construção, execução e compartilhamento de contêineres, cujo processo de implementação de aplicações é chamado de conteinerização. Por possuir em seu *backend* o antigo Linux Container (LXC), compartilha o *kernel* do próprio *host* mantendo o isolamento de cada contêiner, o que não ocorre nas máquinas virtuais (VM's) tradicionais, que necessitam de um S.O. completo e isolado conforme ilustrado na figura 8.

Figura 8 – Docker x VM.



Fonte: https://docs.docker.com/get-started/.

Dessa forma, os contêineres compartilham uma mesma *engine*, possuindo apenas aplicações e bibliotecas dependentes, o que torna a implementação mais leve e possível de serem instaladas em dispositivos de menor poder computacional. Isso aumenta a possibilidade de implantações em ambientes heterogêneos, como o de *fog computing* e da IoT.

Esses não foram os únicos motivos para o incremento desta ferramenta à AFCV. Por ser uma ferramenta de código aberto e uma das mais utilizadas do mercado, permite atender critérios relacionados à interoperabilidade, característica importante para implementações geograficamente distribuídas como é o caso da *fog computing* (MOURADIAN et al., 2018).

Docker Compose:

Por padrão, o Docker Container interrompe o funcionamento de um contêiner quando o mesmo utiliza toda a memória do sistema ou o limite definido no parâmetro. Como a AFCV foi desenvolvida com o intuito de fornecer o gerenciamento de recursos e a continuidade dos serviços para promover a garantia de SLA‘s, esse recurso denominado “oom-killer” foi desabilitado. Em lugar dele, foram utilizados parâmetros no *Compose*, que é um arquivo de extensão YML ou YAML utilizado para definir e executar aplicativos em um ou vários contêineres.

A figura 9 exibe parte do arquivo Docker Compose, onde são apresentados alguns parâmetros de gerenciamento de recursos, dentre os vários possíveis pelo Docker.

Figura 9 – Configuração do Cocker Compose para SLA.

```
plex-server-bruno:
  image: plexinc/pms-docker
  restart: unless-stopped
  container_name: plex-server-bruno
  hostname: plex-server-bruno
  deploy:
    resources:
      limits:
        cpus: '0.50'
        memory: 2G
      reservations:
        cpus: '0.25'
        memory: 500M
  restart: unless-stopped
```

Fonte: Própria do autor.

A AFCV, por ser genérica, é capaz de se adaptar a vários tipos de infraestrutura e contratos de SLA, foram incluídos parâmetros que possibilitam a configuração de acordo com a necessidade do cliente. A figura 9 exemplifica essas possíveis determinações, onde o campo *limits* limita o uso de CPU e memória, nesta ocasião a até 50% e 2 GB respectivamente. Já o *reservations*, está alocando 25% de CPU e 500 MB de memória para o serviço Plex. O campo *restart* se refere à continuidade do serviço, definindo em que condição o mesmo será reiniciado. Na configuração *unless-stopped* o serviço sempre será reiniciado desde que não tenha sido parado manualmente.

Vale ressaltar que apesar de utilizar um *software media center* como gerador de carga, um dos objetivos da AFCV é definir um modelo genérico de SLA adequado aos diversos tipos de infraestrutura de provedores e serviços, disponibilizando recursos para parametrizá-la e monitorá-la. Assim é possível estabelecer uma política de preços para a comercialização da *fog*, pois segundo Alhamad, Dillon e Chang (2010) um ambiente de *fog computing* não pode ser próspero sem um modelo de negócio sustentável.

Docker Swarm:

Outra ferramenta implementada na AFCV é o Docker Swarm, que tem o objetivo de criar *clusters* para os contêineres. Fornecendo à mesma capacidade de escalonamento através de um algoritmo de consenso denominado Raft, o qual também garante questões como a gerência e descoberta de serviços (através dos nós denominados *manager*); redundância da infraestrutura; bem como a alta disponibilidade (através dos serviços fornecidos por todos os nós, sejam eles *managers* ou *works*).

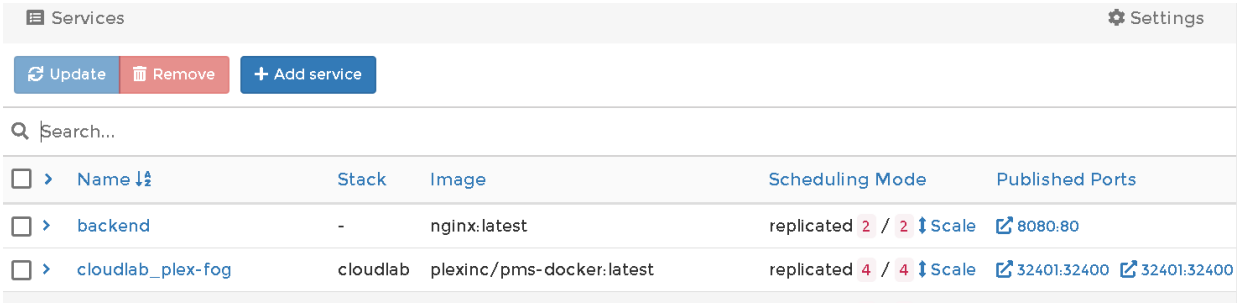
O Swarm foi implementado na camada *fog* deste ambiente considerando seus elementos

essenciais:

• Os **nós**, que são dispositivos físicos aos quais foram atribuídas as nomenclaturas FCN e FN e se referem ao *manager* e aos *workers* respectivamente. O *manager* faz a gestão do *cluster* orquestrando as aplicações, monitorando os nós, cuidando das rotinas internas quando um novo serviço é publicado, recebendo e aplicando atualizações impostas pelo administrador, controlando o ciclo de vida dos contêineres, dentre outras operações. Já os *workers* não possuem nenhuma função administrativa no *cluster*, recebendo apenas as cargas de trabalho vindas do *manager*. Porém, na AFCV, os *workers* foram também utilizados para armazenamento de arquivos, culminando dessa forma numa redução da latência, uma vez que a aplicação buscará esses arquivos localmente ao invés de acessá-los em outro servidor para entregar *streaming* de vídeo aos clientes.

• Os **serviços** no Docker Container são frequentemente uma imagem de um microserviço no contexto de algum aplicativo maior, podendo ser um tipo de programa executável que se deseje executar em um ambiente distribuído. Para a criação de um serviço, pode-se utilizar o arquivo Compose contendo todas as definições do ambiente conforme desejado. Neste ambiente foram implementados 3 serviços: Plex, Nginx e o Portainer. A figura 10 mostra a interface do Portainer exibindo os serviços do Nginx e PLex, bem como as imagens utilizadas, quantidade de tarefas (réplicas) em cada FN e as portas destes serviços.

Figura 10 – Serviços criados na arquitetura.



Fonte: Própria do autor.

• As **tarefas**, são executadas nos nós do *swarm* (nesta ocasião, na *fog*) independentemente umas das outras pelo *manager* após a criação dos serviços. Elas funcionam como um compartimento onde o contêiner é alocado, e caso o mesmo falhe durante as verificações de integridade ou finalize, a tarefa será encerrada. As figuras 11 e 12 exibem os status das 4 tarefas do Plex sendo executadas no FN1 e FN2 respectivamente.

Figura 11 – Tarefas executadas no FN1.

Status	Id	Actions	Slot	Node	Last Update
running	cloudlab_plex-fog.1.jyy36z7dbpr0vgn3odjj3fxkz		1	fog-worker04	2020-03-26 13:05:01
running	cloudlab_plex-fog.2.rv57bilnt9u6y60ykvnwwid7i		2	fog-worker04	2020-03-26 13:05:01
running	cloudlab_plex-fog.3.w9k5ihm34lpml8wu0dn8mtesr		3	fog-worker04	2020-03-26 13:05:01
running	cloudlab_plex-fog.4.0c01kfwokelk3s5t2o007ev14		4	fog-worker04	2020-03-26 13:05:01

Fonte: Própria do autor.

Figura 12 – Tarefas executadas no FN2.

Status	Id	Actions	Slot	Node	Last Update
running	cloudlab_plex-fog.1.jycz7rg00zxrlx63fb6yxrzx5		1	PROCC-236937	2020-03-26 13:05:01
running	cloudlab_plex-fog.2.5s0iu1obujw7s75pjzdmub65y		2	PROCC-236937	2020-03-26 13:05:01
running	cloudlab_plex-fog.3.izzttqtjbbryd93c429j9qvz6		3	PROCC-236937	2020-03-26 13:05:01
running	cloudlab_plex-fog.4.fi9ogprqfhde9lns5nr7aja38		4	PROCC-236937	2020-03-26 13:05:01

Fonte: Própria do autor.

• O **balanceamento de carga** do *swarm* torna os serviços ali criados disponíveis externamente à arquitetura através da publicação de portas, que podem ser definidas manualmente ou automaticamente. Para este ambiente, as portas foram definidas manualmente, preservando o número padrão utilizado por cada aplicação. Os serviços também possuem uma entrada de sistema de nome de domínio (DNS) que é atribuída por um componente DNS interno do *swarm*. Esses nomes são utilizados pelo balanceador de carga para distribuição das solicitações entre serviços dentro do *cluster*.

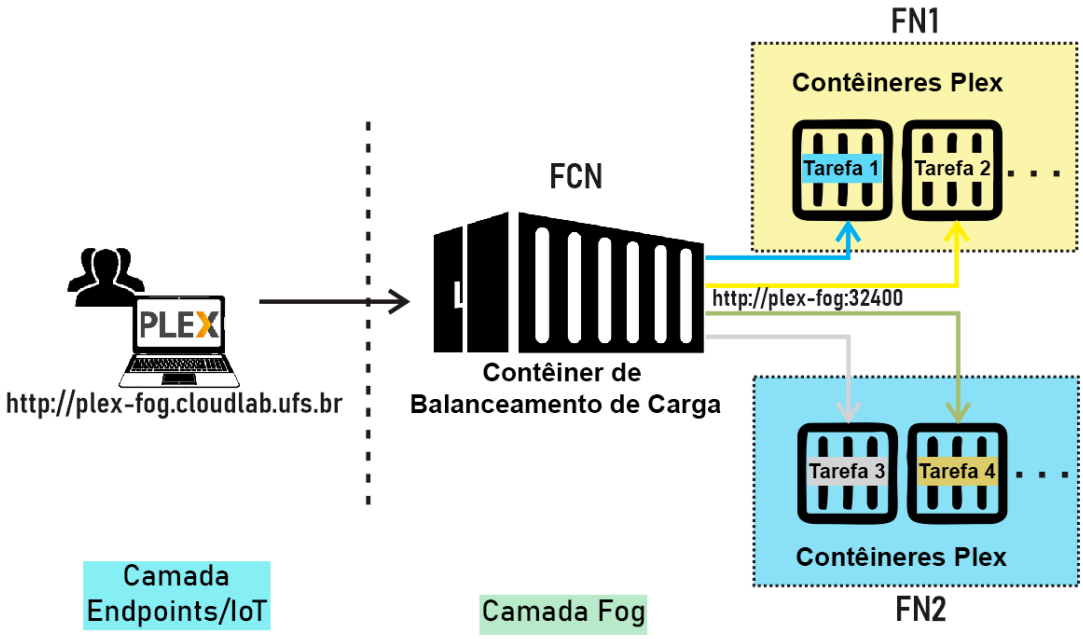
4.2.2.2 Balanceamento de carga com o Nginx

Outra característica da AFCV é prover suporte às mais diversas aplicações, garantindo a melhoria do desempenho, escalabilidade, confiabilidade e do aproveitamento da infraestrutura dos fornecedores do serviço de *fog computing* contribuindo com a QoS. Dessa forma o balanceamento de carga entre os nós é importante para a manutenção de um sistema contínuo, equilibrado e saudável. Mantendo a preocupação quanto à integração entre provedores e permitindo que os mesmos desenvolvam novas formas de negociar os seus serviços, como por exemplo, agregar recursos de clientes à sua infraestrutura conforme sugerido por Alhamad, Dillon e Chang (2010), utilizou-se uma ferramenta de código aberto que atua com diversos protocolos na camada de aplicação, a Nginx (NGINX, 2019).

Como o FCN já possui a função de distribuir os contêineres entre os nós da *fog* na AFCV, o mesmo foi aproveitado também na distribuição de requisições do serviço Plex entre esses contêineres através de um proxy reverso configurado com o Nginx. Como o Docker Container

executa funções de redes virtuais (VNF's), foi criada uma rede *overlay* para a comunicação interna dos contêineres para onde essas requisições são roteadas através de nomes DNS. A figura 13 detalha a implementação do balanceamento de carga nesta prototipação.

Figura 13 – Processo de balanceamento de carga.



Fonte: Própria do autor.

O cliente que está na camada endpoints/IoT, ao acessar o serviço Plex na *fog* através do endereço "plex-fog.cloudlab.ufs.br"configurado na porta 80, será direcionado primeiramente para o FCN que possui o serviço do Nginx instalado, configurado e em execução. Este serviço, através de um algoritmo de balanceamento de carga, fará o redirecionamento para um dos contêineres em execução em algum dos FN's que responderá à requisição do cliente. A partir deste momento o acesso passa a ser feito diretamente entre o cliente e o FN ao qual foi redirecionado, consumindo assim o conteúdo que está armazenado no mesmo. O Nginx permite a parametrização de alguns algoritmos de balanceamento (Least Connections, IP Hash, Least Time, etc.) (NGINX..., 2020), porém foi mantido o Round Robin, que é o padrão utilizado por esta aplicação.

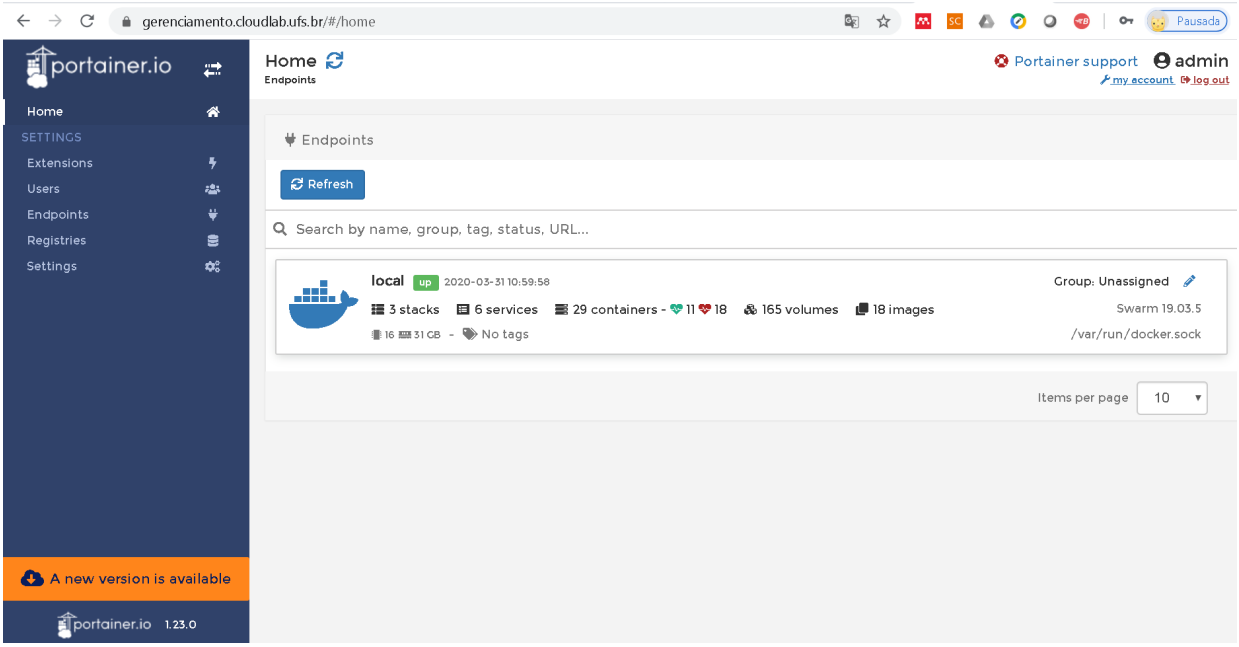
No acesso externo aos serviços, o cliente acessa o CP diretamente através da internet (através do ISP), não fazendo uso do balanceamento de carga. Vale informar que na AFCV os arquivos foram alocados manualmente e de forma individual em cada nó da *fog* (no CP/FCN, no FN1 e no FN2). Apesar disso, todos possuem os mesmos conteúdos.

4.2.2.3 Portainer

O Portainer (PORTAINER, 2018) é uma ferramenta de gerenciamento de código aberto que permite criar, gerenciar e manter ambientes Docker Container. Ele foi integrado à AFCV

por ser um sistema leve e com uma interface gráfica simples, não demandando uma alta curva de aprendizado. Este serviço também foi instalado em um contêiner na VM, onde podia ser acessado internamente (pelo FCN) e externamente (pela CP) através do endereço "https://gerenciamento.cloudlab.ufs.br"conforme apresentado na figura 14. O Portainer também contribui no monitoramento de algumas métricas aferidas neste experimento, conforme veremos a seguir na subseção 4.2.3.

Figura 14 – Tela inicial do Portainer.



Fonte: Própria do autor.

4.2.3 Métricas

As métricas são importantes para demonstrar o desempenho de um ambiente de produção, ou qualquer outro que se precise gerenciar. Na arquitetura de *fog computing*, alguns autores como (MAHMUD; KOTAGIRI; BUYYA, 2016) e (MOURADIAN et al., 2018) descreveram algumas métricas importantes para se gerenciar um ambiente como esse. Também (PRODAN; OSTERMANN, 2009) defende a importância das mesmas para a existência da gerência de contratos de SLA's, necessários para a comercialização desse tipo de serviço.

Apesar de existir várias métricas para aferir um sistema computacional, nesta dissertação foram consideradas as que permitem a análise da aplicabilidade da AFCV, bem como possibilitem a elaboração de contratos de SLA, consolidando toda a estrutura em favor da comercialização de serviços por provedores cuja arquitetura seja baseada ou semelhante à detalhada neste capítulo.

Dessa forma, as métricas consideradas relevantes foram:

• **Consumo de CPU e memória dos dispositivos da *fog*:** estas medidas sinalizam a eficiência dos dispositivos diante de sua capacidade computacional.

• **Largura de banda (transmissão):** utilizada como forma de medir a capacidade de transmissão da rede, ou seja, é o tempo em que um dado é transmitido por completo da origem ao destino. Seu resultado é medido em bit por segundo (bit/s ou bps). Então, se x *bits* são transmitidos em 1 segundo, y *bits* levarão y/x segundo(s) para chegar ao destino.

• **Transferência:** é a quantidade de dados transferidos num intervalo de tempo. Essa métrica é fornecida em bytes (B) e é proporcional à largura de banda. Ela fornece uma visão da capacidade da rede de uma maneira convencional.

• ***Jitter*:** importante métrica para a fog, uma vez que a variação da latência impacta no uso da rede. Representa o desvio padrão do atraso de pacotes enviados em sequência.

• **Perda de Pacotes:** refere-se aos pacotes enviados pelo emissor e não recebidos pelo receptor. Geralmente essa perda se dá pela saturação de enlaces, congestionamento de rede, dentre outros, e tem seu percentual calculado através do total de pacotes não recebidos (perdidos) dividido pelo total de pacotes enviados.

Para se obter os dados referentes à cada métrica na AFCV, houve a necessidade de buscar ferramentas capazes de aferir os dados tanto na camada de aplicação quanto na de transporte, para assim obter informações de todo o sistema e respectiva topologia. Para isso, foi utilizada uma aplicação para fornecimento do serviço, gerando carga na AFCV 4.2.4. Os testes e dados obtidos foram possíveis com o auxílio da ferramenta Blazemeter 4.2.4.1. Para a outra análise utilizou-se a ferramenta Iperf 4.2.4.2.

4.2.4 Carga de Trabalho

O centro de mídia é uma ferramenta que vem sendo muito utilizada no fornecimento de *streaming* de vídeo e armazenamento de dados, funcionalidades estas também integrantes de ambientes inteligentes, como o de *smart home* por exemplo. Por já estar disponível em forma de contêiner na plataforma Github (GITHUBPLEX, 2019) e possuir licença *open-source*, foi escolhida a ferramenta Plex (PLEX, 2019). O intuito foi de injetar carga na AFCV através do fornecimento de seus serviços os quais condizem com o propósito da mesma. Existem outras soluções responsáveis por entregar serviços semelhantes, como a Streama (STREAMA, 2019), Red5 (RED5, 2019), Wowza (WOWZA, 2019), etc, porém, pelas características já descritas, o Plex demonstrou melhor aplicabilidade. Há também arquiteturas específicas para esse tipo de aplicação, como a utilizada pela Netflix, que utiliza *appliances* próprias denominadas de *Open Connect Appliances* (OCA's) distribuídas em diversos provedores locais, formando uma rede de entrega de conteúdo (CDN). Mesmo dispondo dessa estrutura, esta empresa é responsável pelo maior consumo *downstream* da Internet (19,10%), sendo também responsável por até mais de 40% da banda de provedores locais em horários de pico, segundo a Sandvine (2018).

Conforme abordado no tópico motivação, onde citamos a previsão de crescimento da quantidade de dados gerados pela IoT (ambiente este no qual o *streaming* de vídeo se insere)

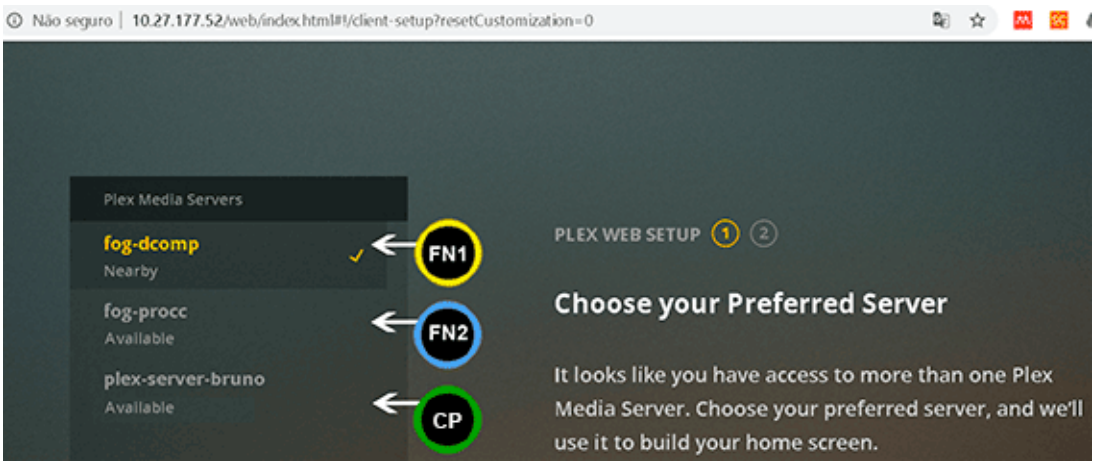
gera-se a necessidade de buscar alternativas que venham beneficiar e garantir aos clientes e provedores uma melhor experiência e economia, uma vez que não se trata de uma demanda futura, mas atual.

Este tipo de aplicação necessita de um ambiente de baixa latência para a execução contínua e satisfatória de seu conteúdo. Segundo o artigo do próprio desenvolvedor do Plex (WHAT..., 2019), quando utilizado localmente, ele raramente precisará de transcodificação (indicando um melhor desempenho num ambiente de *fog computing*). Com isso, há uma significativa economia de recursos de processamento.

O Plex fornece seus serviços de *streaming* e armazenamento através dos contêineres que compõem a AFCV conforme detalhado anteriormente, os quais são alocados em cada nó da *fog* e acessados por um cliente através de um navegador que exibe sua interface gráfica. Nesta interface é possível visualizar e até mesmo escolher de qual servidor a aplicação irá buscar os dados. A aplicação já sugere aquele que está mais próximo, ou seja, quando acessado através da camada *endpoint*, a mesma irá selecionar automaticamente o servidor da *fog* ao qual o cliente foi redirecionado pelo balanceador de carga.

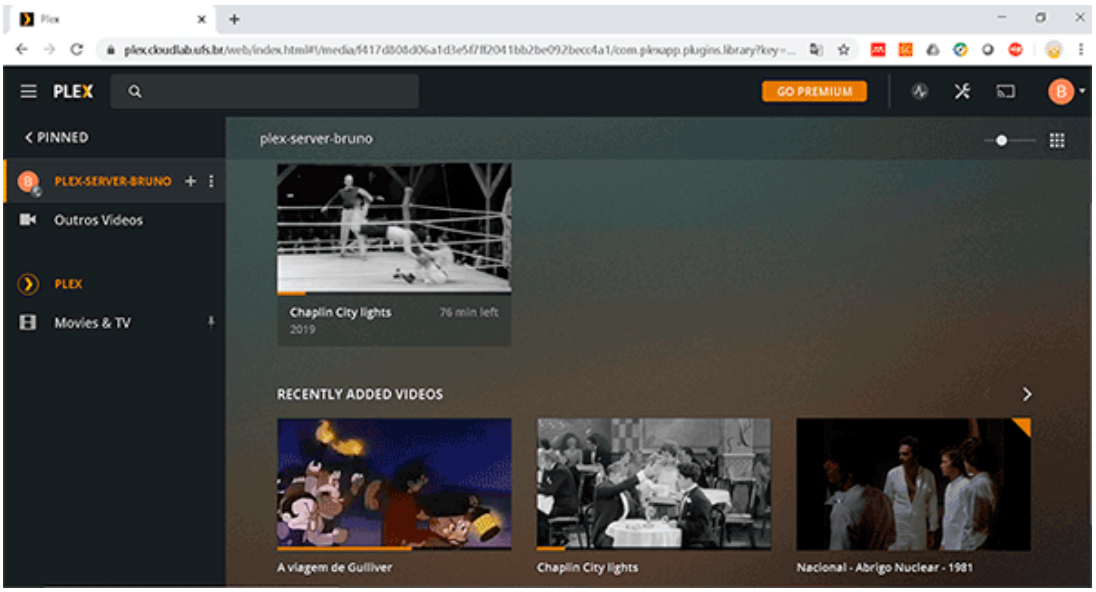
A figura 15 exibe essa interface com os 3 servidores implementados e disponíveis (fog-dcomp, fog-procc e plex-server-bruno), que foram identificados através dos círculos coloridos contendo suas respectivas siglas (FN1, FN2 e CP). É possível também visualizar que o servidor selecionado automaticamente (fog-dcomp) possui um status ("*Nearby*") logo abaixo de seu nome, indicando a questão da proximidade comentada acima. É possível comprovar o funcionamento deste serviço pela presença do IP (10.27.177.52) na barra de endereço do navegador, o qual pertence de fato ao FN1. Isso não impede de acessar o conteúdo de outro servidor. Basta que o *status* do mesmo esteja como "*Available*"(disponível) para que se possa selecioná-lo, e a partir daí a aplicação passará a exibir e executar o conteúdo deste servidor. A figura 16 apresenta a tela de conteúdos do CP acessado através do endereço "plex.cloudlab.ufs.br".

Figura 15 – Interface do Plex para escolha do servidor.



Fonte: Própria do autor.

Figura 16 – Tela de conteúdos do Servidor Plex CP.



Fonte: Própria do autor.

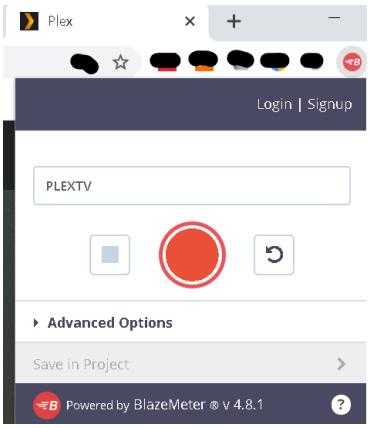
Neste ambiente, os arquivos disponibilizados na central de mídia foram inseridos em cada servidor manualmente. Vale ressaltar que foram utilizados arquivos de vídeo de domínio público, a fim de evitar problemas relacionados à direitos.

4.2.4.1 BlazeMeter

O BlazeMeter (BLAZEMETER, 2019) é uma plataforma de testes de performance para aplicações. Como foi implementado um serviço de central de mídia fornecido através do *software* Plex, se faz necessário aferir esta carga gerada. O objetivo é saber o comportamento da AFCV e seus dispositivos durante o acesso ao serviço mídia pelos clientes.

A escolha desta ferramenta deve-se ao fato dela reunir várias funcionalidades, como a gravação da utilização de um serviço *web* (como o Plex) para a geração de um *script*. Este pode ser alterado para a diversificação dos testes, como alterar a quantidade de acessos simultâneos de usuários (limitado a 50 na versão gratuita), por exemplo. Isso é importante para a obtenção de um maior tráfego neste ambiente para aferição de aspectos relacionados à QoS e o impacto na AFCV sob uma maior demanda. Para esta gravação, foi utilizado um *plugin* denominado BlazeMeter Chrome Extension (figura 17).

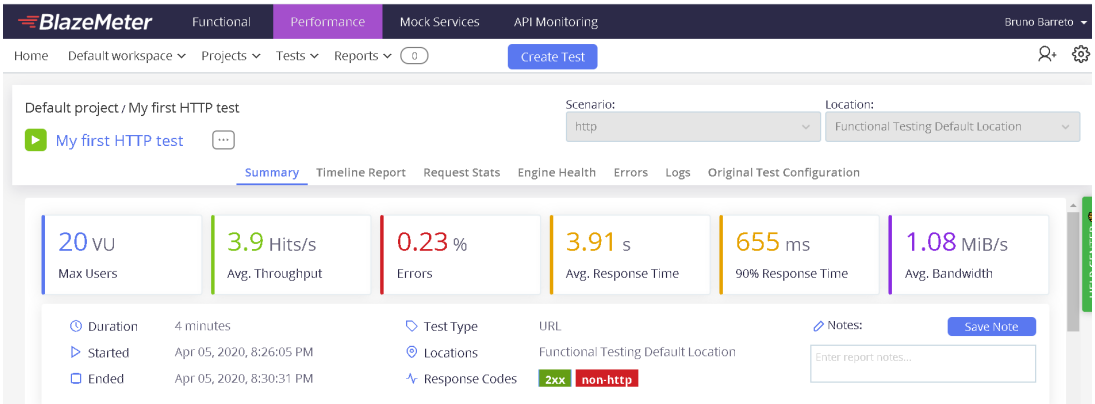
Figura 17 – Plugin do BlazeMeter para o navegador Chrome.



Fonte: Própria do autor.

O BlazeMeter também permite análises abrangentes através de relatórios em tempo real, incluindo métricas como largura de banda, latência, erros, dentre outras. Estes relatórios permitem uma melhor avaliação da AFCV na perspectiva de uma utilização de serviços em ambiente real, contribuindo para o aperfeiçoamento da análise da QoS e auxiliando na comprovação do fornecimento do serviço acordado com o cliente através do contrato de SLA. A figura 18 apresenta a tela de resultados após um teste realizado com essa ferramenta.

Figura 18 – Tela de resultados de teste do BlazeMeter.



Fonte: Própria do autor.

4.2.4.2 Iperf

Ferramentas como o Iperf (IPERF, 2019) são utilizadas em ambientes como este, a fim de avaliar a topologia através da injeção e análise de dados e pontos críticos da rede, permitindo verificar por exemplo se a QoS está conforme desejado e até mesmo auxiliando no planejamento futuro da rede.

O Iperf é considerado uma ferramenta de medição ativa, pois seu funcionamento se dá através da injeção de fluxos de dados na rede. A partir de dois dispositivos (1 cliente e 1 servidor)

pode-se avaliar o comportamento desse tráfego e assim medir o desempenho da mesma. Esses fluxos podem ser gerados com características específicas, simulando o acesso a uma aplicação como a de vídeo, por exemplo. Ele atua na camada de transporte através dos protocolos TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*) e (*Secure Copy Protocol*), e possibilita a obtenção de algumas métricas, inclusive 3 das definidas na subseção 4.2.3: largura de banda fim a fim, o *jitter* e a perda de pacotes.

A figura 19 exibe a execução de dois testes com a ferramenta Iperf entre o cliente FN1 e o servidor FCN. O primeiro utilizando o protocolo TCP e o outro com o UDP. Percebe-se nos resultados que o Protocolo UDP fornece 2 métricas a mais que o TCP, que são o *jitter* e a perda de pacote (datagrama).

Figura 19 – Testes entre o FN1 e o FCN com o Iperf utilizando os protocolos TCP e UDP.

```
root@Cloudlab:/home/administrator# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.27.196.248 port 5001 connected with 10.27.177.52 port 47688
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.1 sec   113 MBytes  94.0 Mbits/sec
^Croot@Cloudlab:/home/administrator# iperf -s -u
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.27.196.248 port 5001 connected with 10.27.177.52 port 38156
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec   0.044 ms    0/  892 (0%)
```

Fonte: Própria do autor.

5

Resultados e Discussão

Este capítulo apresenta os detalhes dos experimentos realizados na AFCV implementada, assim como a análise de seus resultados através de métricas como as relacionadas na seção 4.2.3, além de outras obtidas com as ferramentas Iperf (4.2.4.2) e Blazemeter (4.2.4.1).

5.1 Detalhamento dos Experimentos

Conforme já mencionado, a AFCV foi implementada utilizando a infraestrutura da UFS, e assim como num ambiente comum de provedores de serviços, vários tipos de tráfegos se movem por sua infraestrutura a todo momento. Como essa infraestrutura não possui um isolamento de fluxo para a AFCV, o serviço utilizado nessa arquitetura acaba competindo com os demais, e dessa forma se faz necessário reunir evidências concretas (números) sobre o comportamento da mesma antes (sem gerar carga) e durante a execução de qualquer teste de desempenho.

Diante dos dispositivos convencionais utilizados, e após a implementação de todos os serviços necessários ao propósito da AFCV, conduzimos primeiramente a análise sem gerar carga de teste, apenas com o que está sendo consumido pelos serviços vitais da AFCV, para identificar a utilização de CPU, o consumo de memória e o tráfego de rede individualmente nos dispositivos CP/FCN, FN1 e FN2. Essa análise é importante para um ambiente de *fog computing*, pois geralmente seu conjunto é formado por dispositivos com limitações de recursos. Segundo Jain (1991), essa é uma das formas de expressar esses consumos.

Os testes com carga foram realizados tanto da camada endpoints/IoT para a nuvem (através do ISP) quanto para a *fog* (fazendo uso da AFCV). Isso permite comparar o modelo de nuvem tradicional com a AFCV e assim identificar o ambiente mais apropriado para cada tipo de aplicação segundo seus requisitos. Dessa forma, também possibilita comprovar se de fato a AFCV é capaz de atender ao objetivo proposto nesta dissertação.

A fim de obter uma medição mais assertiva, os testes foram realizados 1 por vez com

o objetivo de evitar concorrência entre eles, preservando apenas o trafego convencional da infraestrutura utilizada. Para coletar estas evidências, consideramos o tempo de duração dos vídeos utilizados (que variam entre 10 a 18 minutos) para a realização dos testes da seção 5.4, assim pudemos garantir que nenhum deles terminasse sua execução antes de finalizarmos a coleta dos dados.

Dessa maneira, monitoramos o comportamento de cada nó em tempo real durante 10 minutos, considerando a média dos intervalos de 60 segundos, ou seja, eram obtidas 10 médias para a amostra. Esse tempo foi utilizado como padrão para a coleta das amostras utilizadas em todas as análises realizadas nas próximas seções.

Em relação aos gráficos apresentados neste capítulo, foram seguidas algumas orientações fornecidas por Jain (1991). Segundo esse autor, os gráficos devem apresentar as informações de maneira a exigir o mínimo de esforço do leitor, e para a escolha do melhor modelo de representação, é importante saber identificar o tipo de variável que irá apresentar. Devido a possibilidade de ocorrerem picos de consumo de recursos nos dispositivos, foi escolhido o gráfico do tipo *box plot*, pois possibilita avaliar rapidamente a dispersão das métricas e também os possíveis valores discrepantes (*outliers*) presentes.

A seguir na seção 5.2 apresentamos os resultados da análise sem a geração de carga de teste. Em seguida, na seção 5.3 são apresentados os resultados do tráfego gerado com a ferramenta Iperf (4.2.4.2) considerando o ponto de vista das aplicações com os protocolos TCP e UDP. Por fim, com o auxílio da ferramenta Blazemeter (4.2.4.1) analisamos na seção 5.4 a utilização de recurso na execução do serviço central de mídia.

5.2 Análise do Consumo de Recursos da AFCV Sem Carga de Teste

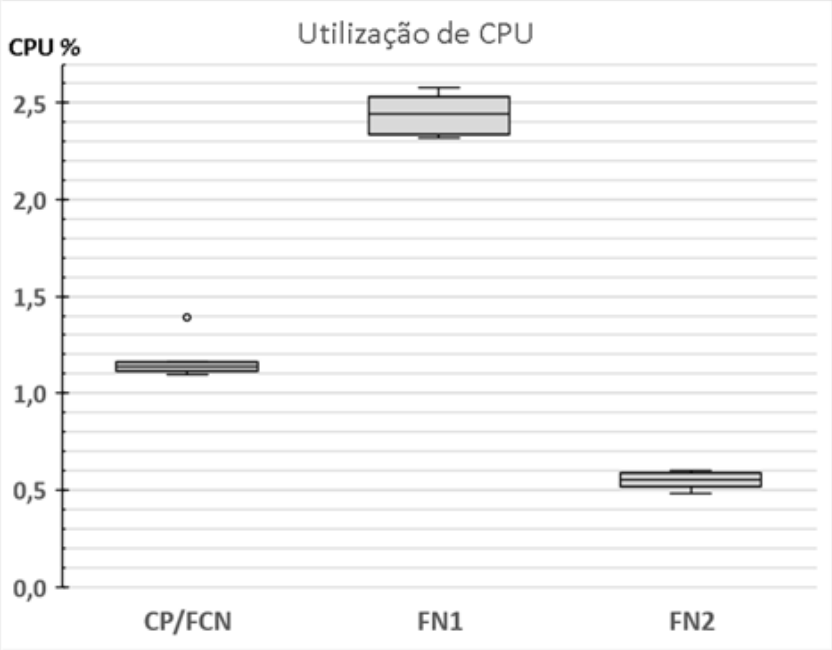
Os gráficos *box plot* utilizados nessa análise representam a utilização de CPU, memória e rede após a implementação e configuração da arquitetura para a execução dos testes. Durante essa medição estavam implementados além dos *softwares* básicos para o funcionamento de um sistema computacional, os serviços virtualizados e clusterizados com o Docker Container.

O fato de se utilizar um ambiente real no experimento pode ser testemunhado logo no início das primeiras medições, onde mesmo sem a geração da carga na arquitetura os dispositivos já apresentam consumos bem distintos e correspondentes à característica e capacidade de cada um.

O gráfico ilustrado na figura 20 representa a utilização de CPU pelos dispositivo da arquitetura. O FN1 foi o dispositivo que apresentou maior consumo de CPU após a implementação da AFCV, possuindo uma média de 2,45% de utilização com um pico máximo de 2,6% e mínimo de 2,32%. O CP/FCN é o segundo dispositivo a apresentar maior uso de CPU mesmo possuindo um

maior número de serviços implementados. Devido ao seu alto poder computacional a utilização ficou em média 1,14% havendo um *outlier* de 1,39% que se deu no momento em que a interface de gerenciamento do Portainer foi acessada. Após essa interface já aberta repetimos a análise mais duas vezes e não houve o aparecimento de *outlier*. O FN2 foi o dispositivo que menos consumiu CPU, tendo comprometido uma média de 0,56% e atingindo um pico máximo de 0,60% e mínimo de 0,48%, sendo assim a que menos sentiu o impacto causado pela virtualização.

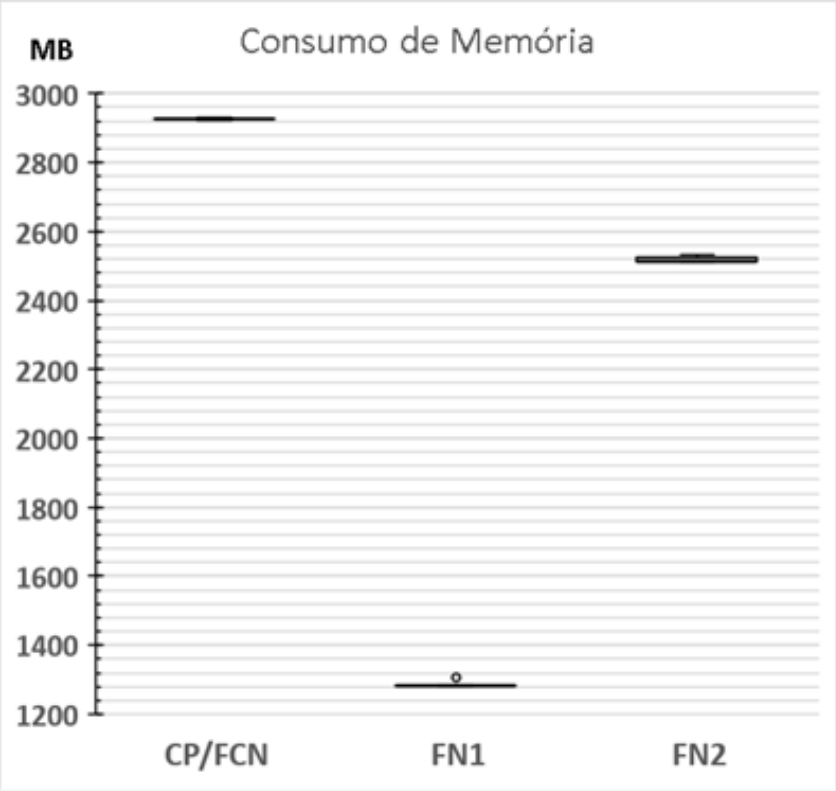
Figura 20 – Utilização de CPU após implementação da AFCV.



Fonte: Própria do autor.

Diferente do que foi visto em relação ao consumo de processamento, as caixas do gráfico de uso de memória, apresentadas na figura 21, possuem menor amplitude (maior valor menos o menor) e um pequeno *outlier* no FN1. O CP/FCN apresenta uma média maior de utilização de memória (2927 MB) do que os outros dispositivos, causada pela quantidade de serviços que ele fornece. O FN1 possui uma versão de S.O. mais enxuta (sem interface gráfica) do que o FN2, e por isso consome menos memória (média de 1283 MB). Em relação ao *outlier* ocorrido, apesar de representar um desvio de apenas 22 MB em relação à média, analisamos mais duas amostras para detectar possíveis reincidências, porém isso não ocorreu. Já o FN2 utiliza uma versão de S.O. com interface gráfica, o que acaba aumentando a demanda de memória, chegando este dispositivo a consumir uma média de 2516 MB. Vale lembrar que além do S.O., todos eles possuem o Docker Container instalado e com os serviços disponíveis. A utilização deste recurso com os serviços implementados da AFCV implementados é considerada baixa, já que não chega a 3 GB em nenhum dos dispositivos. Isso representa a viabilidade de instalação em dispositivos comuns.

Figura 21 – Utilização de memória após implementação da AFCV.

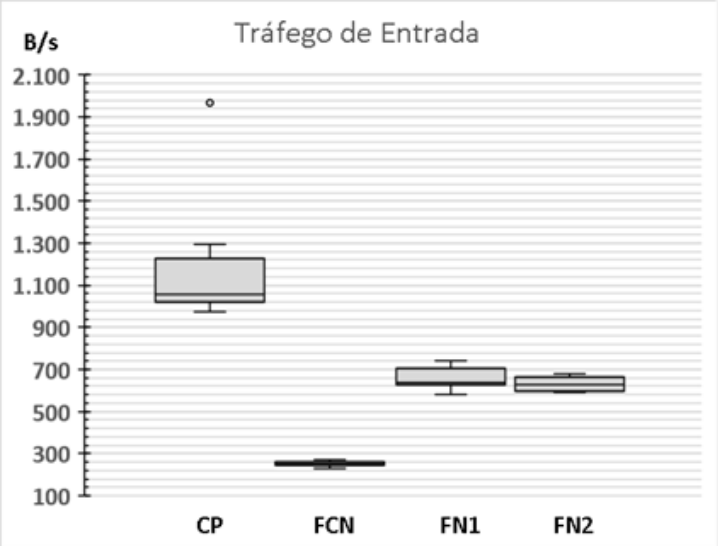


Fonte: Própria do autor.

Conforme mencionamos no capítulo 4 (seção 4.2.2), sempre que forem abordados os resultados referentes a aspectos de rede, iremos nos referir à VM (CP/FCN) separadamente, onde a CP fará referência à comunicação externa (da nuvem) e o FCN à interna (da *fog*). Além disso, o tráfego de dados na rede foi separado em tráfego de entrada (*download*) e tráfego de saída (*upload*).

A figura 22, que representa o tráfego de entrada, indica uma utilização média de 1050 B/s na interface do CP com um valor discrepante de 1960 B/s. O FCN possui uma média de 250 B/s, e inversamente ao CP, representa o menor valor dentre eles. Os FN1 e FN2 apresentam médias semelhantes, 640 e 625 B/s respectivamente. O fato do CP possuir o maior tráfego dentre as interfaces e ocorrer *outlier* está relacionado a sua exposição à Internet, que por ser um ambiente mais difícil de ser controlado, acaba existindo esta desvantagem se compararmos com os dispositivos da *fog*, já que nessa camada temos um maior controle do que trafega internamente.

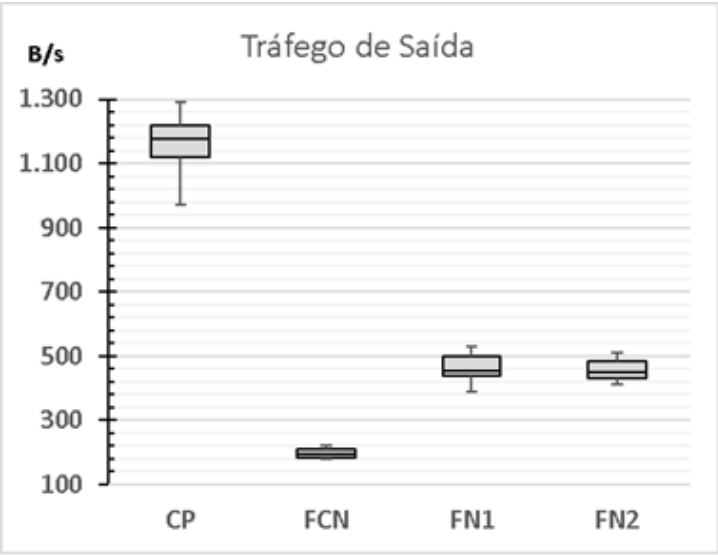
Figura 22 – Tráfego de entrada após implementação da AFCV.



Fonte: Própria do autor.

O tráfego de saída ilustrado na figura 23 segue de forma análoga ao de entrada, proveniente da quantidade de requisições recebidas, porém com pequenas diferenças entre os valores e com a inexistência de *outliers*. O dispositivo CP obteve o maior volume de dados entre os demais dispositivos, com uma média de 1150 B/s. O FCN seguiu com a menor média (195 B/s), e os dispositivos FN1 com 455 B/s e FN2 com 450 B/s.

Figura 23 – Tráfego de saída após implementação da AFCV.



Fonte: Própria do autor.

5.3 Análise do Tráfego Gerado do Ponto de Vista das Aplicações com os Protocolos TCP e UDP

Conforme nossa preocupação em prover uma arquitetura genérica capaz de atender a diferentes demandas, além do protocolo da camada de transporte TCP utilizado pela aplicação escolhida neste trabalho para geração de carga, foram incluídos testes com o protocolo UDP pertencente à mesma camada e comumente utilizado em aplicações menos exigentes quanto à entrega dos dados (menos sensíveis).

Os testes realizados com a ferramenta Iperf (4.2.4.2) vão além de verificar a conectividade de rede (conseguida com a ferramenta Ping) e a alcançabilidade (através da Traceroute). Eles buscam estressar a rede transferindo o máximo de dados (*upload* e *download*) num intervalo de tempo como se fosse uma aplicação para obtermos algumas métricas como largura de banda, *jitter*, perdas de pacotes, etc.

Primeiramente foi preciso testar o funcionamento e a capacidade de cada dispositivo por meio da execução do servidor e do cliente na mesma máquina utilizando o endereço de *loopback* (127.0.0.1) e o protocolo TCP. Isso permitiu verificar internamente sua capacidade de transferência de dados e largura de banda, identificando possíveis limitações.

Os resultados apresentados na tabela 5 informam que os dispositivos são suficientemente capazes de atuar na AFCV, pois internamente atingem uma média superior a 27 Gb/s de largura de banda e conseguem transferir mais de 1800 GB de dados num intervalo de 10 minutos (tempo utilizado na realização testes), valores esses muito acima dos limites impostos pela infraestrutura utilizada (100 Mb/s da rede interna (LAN) e de 50 Mb/s da Internet (WAN)).

Tabela 5 – Capacidade interna de comunicação.

	Transferência GB	**Largura de Banda Gb/s**
CP/FCN	2230	31,9
FN1	1893	27,1
FN2	2157	30,9
EP1	2180	31,2
EP2	2164	31,1

A seguir, apresentamos os resultados das métricas obtidas através das comunicações fim a fim realizadas com os protocolos TCP e UDP entre o *endpoint* (EP1) e a nuvem privada (CP), bem como entre o EP1 e cada nó da *fog* (FCN, FN1 e FN2). Devido à quantidade de dados existente neste tipo de experimento, há a necessidade de se calcular a dispersão dos mesmos da média através do desvio padrão da amostra coletada (JAIN, 1991), e por isso esta métrica foi incluída nos resultados. Também são apresentados os valores mínimos e máximos de cada intervalo capturado e a média dos 10 minutos para as métricas de transferência (MB) e largura de banda (Mb/s). A média do *jitter* (ms) e o total de pacotes perdidos (%) também foram analisados

através do protocolo UDP.

Na tabela 6, são apresentados os resultados referentes à estrutura da nuvem convencional na qual o EP1 se comunica diretamente com o dispositivo da nuvem privada (CP). Primeiramente apontaremos as vantagens do protocolo UDP sobre o TCP em todos os testes, onde transferiu uma média de 152,6 MB contra 53,76 MB do TCP, e atingiu uma largura de banda de 21,33 Mb/s contra 7,5 Mb/s. Isso já era de se esperar devido às características dos mesmos, uma vez que o TCP é orientado à conexão e por isso faz maiores exigências para garantir a confiabilidade da mesma. Essa confiabilidade do TCP é perceptível pelos menores valores dos desvios padrões, que indicam uma maior estabilidade na comunicação.

Tabela 6 – Resultados da comunicação com a nuvem.

Resultados entre o Endpoint (EP1) e a Nuvem Privada (CP)					
		MÍN	**MÁX**	**MÉD**	**DES**
TCP	**Transferência (MB)**	3,63	71,6	53,76	19,69
	Largura de Banda (Mb/s)	0,5	10	7,5	2,75
UDP	**Transferência (MB)**	103	238	152,6	43,67
	Largura de Banda (Mb/s)	14,4	33,3	21,33	6,1
	Jitter (ms)	1,096			
	Pacotes Perdidos (%)	16			

Na tabela 7 apresentamos os resultados referentes à comunicação entre o EP1 e os dispositivos da *fog* (FCN, FN1 e FN2) para assim podermos comparar com os resultados da nuvem. Em relação aos protocolos TCP e UDP os resultados demonstram as mesmas vantagens já apresentadas na nuvem, sendo o UDP com melhores taxas de transferência e maior largura de banda, e o TCP mais estável com menor desvio padrão em todos os dispositivos.

Tabela 7 – Resultados da comunicação com a Fog.

Resultados entre o Endpoint (EP1) e o Fog Control Node (FCN)					
		MÍN	**MÁX**	**MÉD**	**DES**
TCP	**Transferência (MB)**	676	678	677,6	0,7
	Largura de Banda (Mb/s)	94,5	94,9	94,75	0,11
UDP	**Transferência (MB)**	679	685	683,9	1,79
	Largura de Banda (Mb/s)	94,9	95,7	95,58	0,24
	Jitter (ms)	0,099			
	Pacotes Perdidos (%)	0			
Resultados entre o Endpoint (EP1) e o Fog Node 1 (FN1)					
		MÍN	**MÁX**	**MÉD**	**DES**
TCP	**Transferência (MB)**	676	679	678,3	1,06
	Largura de Banda (Mb/s)	94,5	94,9	94,8	0,14
UDP	**Transferência (MB)**	681	685	683,7	1,06
	Largura de Banda (Mb/s)	95,1	95,7	95,6	0,19
	Jitter (ms)	0,18			
	Pacotes Perdidos (%)	0			
Resultados entre o Endpoint (EP1) e o Fog Node 2 (FN2)					
		MÍN	**MÁX**	**MÉD**	**DES**
TCP	**Transferência (MB)**	677	679	678	0,47
	Largura de Banda (Mb/s)	94,7	94,9	94,8	0,05
UDP	**Transferência (MB)**	681	684	683,6	0,97
	Largura de Banda (Mb/s)	95,3	95,7	95,63	0,13
	Jitter (ms)	0,098			
	Pacotes Perdidos (%)	0			

Comparando a comunicação entre a nuvem e a *fog* através das tabelas 6 e 7, percebe-se que na *fog* além de obtermos valores muito melhores em relação à nuvem eles também são mais homogêneos, ou seja, em todos os nós da *fog* temos valores aproximados com os dois protocolos, o que comprova um melhor desempenho e consistência na comunicação da AFCV. O baixo desvio padrão, *jitter* e perda de pacotes também contribuem para uma melhor disponibilidade dos serviços e possibilitam uma melhor QoS.

Comparando os valores obtidos na nuvem com os maiores valores da AFCV, percebemos um desvio padrão máximo de 43,67 MB de transferência de dados com o protocolo UDP, enquanto que na AFCV o maior desvio padrão foi de 1,79 MB no dispositivo FCN. O *jitter* foi de 1,096 ms no CP contra 0,18 ms nos testes com o dispositivo FN1. Já a perda de pacotes no CP foi de 16%, enquanto na *fog* não tivemos perda. Esses valores comprovam tanto as deficiências da nuvem, capaz de impactar em aplicações mais sensíveis, como mostra a aptidão da AFCV em suprir essa demanda.

Devido à diferença de capacidade entre o *link* utilizado para acessar a nuvem (50 Mb/s) e a fog (100 Mb/s) e considerando as diferenças entre os protocolos de transporte utilizados, apresentamos na tabela 8 a eficiência da comunicação entre os dispositivos. Segundo Jain (1991) essa medida se dá pela divisão entre a capacidade utilizável sobre a nominal, o que nos permite analisar o aproveitamento dos recursos de rede de forma equivalente.

Tabela 8 – Eficiência de comunicação entre os dispositivos.

Testes entre o Endpoint (EP1) e –>		**CP**	**FCN**	**FN1**	**FN2**
Eficiência %	**TCP**	15.02	94.75	94.81	94.80
	UDP	42.66	95.58	95.60	95.63

De acordo com a tabela 8, a AFCV também utiliza melhor os recursos de rede. O menor aproveitamento foi entre o EP1 e o FCN com o protocolo TCP (94,75%), e o maior foi com o FN2 usando UDP (95,63%). Com a CP obtivemos um aproveitamento mínimo de 15,02% com o TCP e máximo de 42,66% com o UDP. Esses resultados nos mostram a importância dos dispositivos estarem mais próximos da borda (isto é, mais próximo dos dispositivos finais) como acontece na AFCV, obtendo um ganho em nossos experimentos sobre a nuvem de 52,97% com o UDP e 79,72% com o TCP.

5.4 Análise da Utilização de Recurso na Execução do Serviço Central de Mídia

Para as medições das cargas geradas através do serviço web de central de mídia (Plex), a ideia inicial foi realizar apenas um teste em cada servidor abrangendo o acesso, a navegação e a execução dos vídeos na central de mídia através da ferramenta Blazemeter, porém, essa ferramenta não foi capaz de capturar as informações geradas a partir do *player* do aplicativo Plex por esse ser executado numa sobrecamada (*overlay*).

Diante disso, foi necessário realizar 2 testes com o serviço central de mídia. Primeiramente realizamos o teste de performance relacionado ao acesso e navegação do cliente à central de mídia, o qual denominamos de **Teste_1**. Esse teste foi realizado através da ferramenta Blazemeter que foi acessada no cliente EP1 e parametrizada em sua capacidade máxima de injeção de dados respeitando os limites impostos pela versão gratuita (50 usuários simultâneos). Além da duração e da quantidade de usuários, também foram efetuados testes de escalabilidade. Para isso, foi definido que os 50 usuários acessariam de forma gradativa durante os primeiros 5 minutos de testes, onde 10 usuários acessavam a cada minuto, obtendo o máximo de acessos a partir do 5º minuto.

Uma vez gravados e parametrizados os *scripts* com os acessos e navegação pelo serviço de central de mídia disponibilizado por cada servidor (CP, FN1 e FN2), o mesmo foi executado para a geração da carga nos ambientes de nuvem e *fog*.

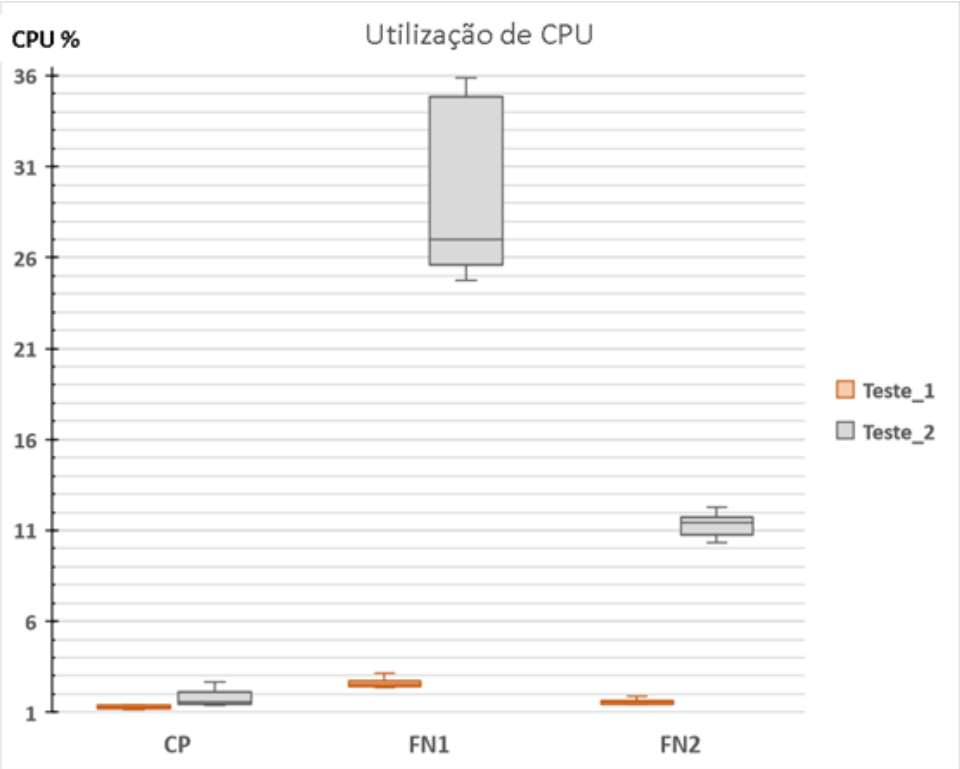
Para a avaliação do consumo de recursos relacionados à execução de vídeos através da central de mídia, foi realizado um segundo teste (**Teste_2**) utilizando como cliente os dois *endpoints* mencionados na AFCV (EP1 e EP2), nos quais foram realizados os acessos à 10 vídeos cada, 5 utilizando o navegador Google Chrome e os outros 5 no Microsoft Edge. Os testes realizados na nuvem se deu através do dispositivo CP que forneceu os 20 vídeos por ser o único

servidor deste serviço nesta camada. Já os da *fog*, por possuir o recurso de balanceamento de carga implementado 4.2.2.2, as requisições eram redirecionadas de forma igualitária diretamente aos dispositivos FN1 e FN2, resultando numa entrega de 10 vídeos à cada FN.

Realizados os dois testes detalhados na seção 5.1, comparamos a utilização de recursos por cada um deles a fim de analisarmos o impacto nos dois ambientes e sabermos suas respectivas capacidades. Sendo assim, iniciamos a análise a partir do consumo de CPU apresentado na figura 24.

O consumo de CPU aumentou durante a execução dos dois testes em relação ao que vinha sendo consumido pelos dispositivos sem a carga 5.1. No Teste_1, percebemos um aumento pouco significativo, em que os dispositivos tiveram um consumo médio de 1,27%, 2,6% e 1,57% para os dispositivos CP, FN1 e FN2 respectivamente. Devido à necessidade de transcodificação de vídeo em sua execução, ainda que em menor proporção em relação à nuvem conforme (WHAT. . . , 2019), o aumento do uso de CPU no segundo teste foi maior. Esse aumento foi mais sentido nos dispositivos da *fog* (FN1 e FN2), pois além de possuírem recursos mais limitados quando comparados com o CP, eles receberam um volume maior de dados como veremos na análise de rede mais adiante. Porém, todos foram capazes de atender à demanda com a mesma qualidade. Destacamos uma maior variação do processamento (observado pela amplitude da caixa) no dispositivo FN1. Esse dispositivo também recebeu o maior volume de dados entre todos, e já era o dispositivo mais impactado nesse recurso pela virtualização.

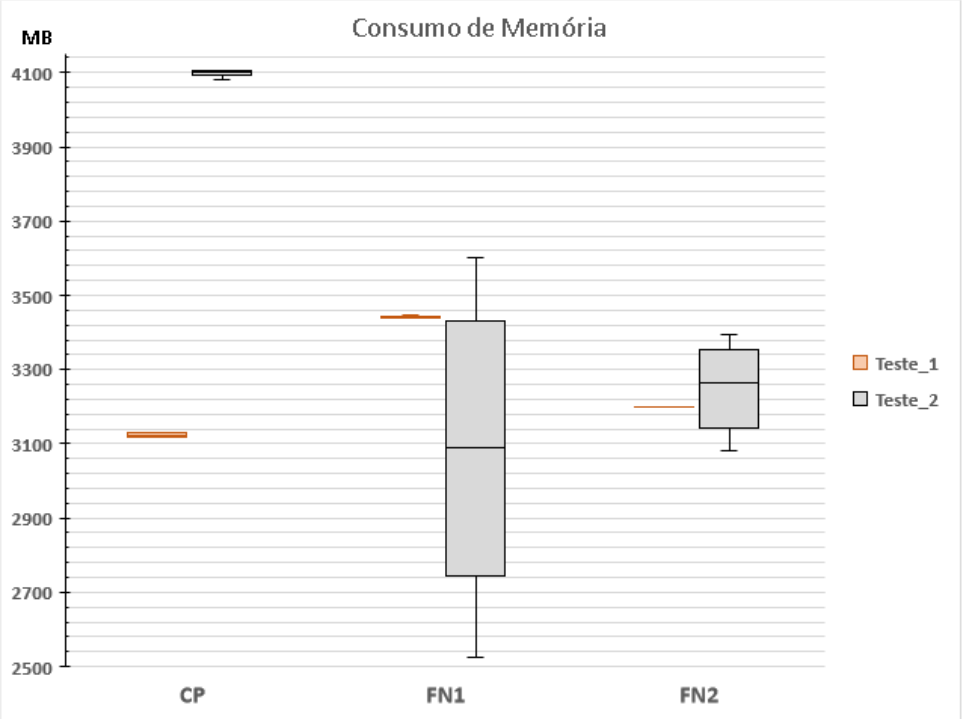
Figura 24 – Utilização de CPU durante os Testes 1 e 2.



Fonte: Própria do autor.

A figura 25 permite compararmos o uso de memória nos diferentes dispositivos e camadas. Nos dois testes como já era de se esperar, houve o aumento no consumo de memória dos dispositivos em relação ao consumo sem a carga. No Teste_1, esse aumento foi de 6,85% no CP, 168,75% no FN1 e 27,49% no FN2. No Teste_2 esse aumento significou 40,40% no CP, 140,62% no FN1 e 29,48% no FN2.

Figura 25 – Utilização de memória durante os testes 1 e 2.



Fonte: Própria do autor.

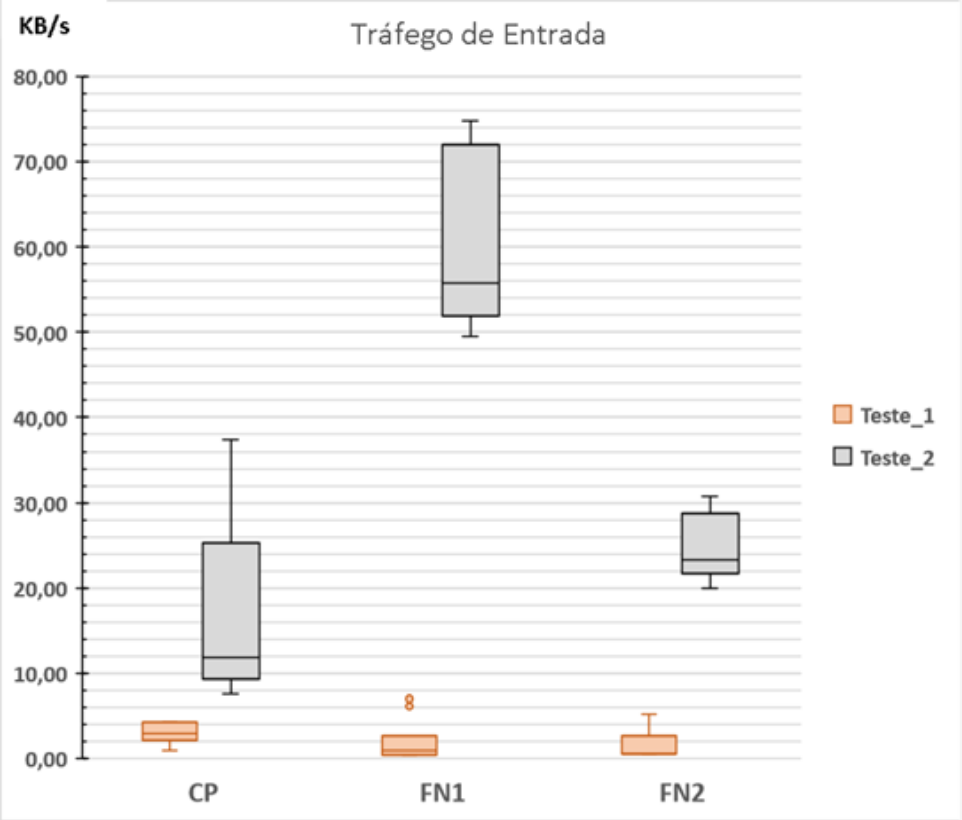
O dispositivo FN1 apresentou os maiores consumos de memória no Teste_1, consumindo 3443 MB, valor mais alto que o consumido no Teste_2 (3083 MB). Nos outros dispositivos o consumo se apresentou de forma inversa, onde o CP utilizou 3123 MB e 4098 MB e o FN2 3199 MB e 3248 MB no primeiro e segundo teste respectivamente. O FN2 foi o dispositivo que menos variou o consumo de memória de um teste para o outro (1,99%), e o CP o que mais variou (33,55%), mostrando que o impacto neste recurso pela utilização de vídeos através da nuvem é bem mais significativo do que na *fog*.

Outra questão a ser observada em relação ao consumo de memória, se deve ao fato dele aumentar com o passar do tempo de execução do Teste_2. Isso ocorreu em todos os nós, porém, essa variação é melhor representada na figura 25 pelos dispositivos FN1 e FN2, onde a amplitude interquartil (maior comprimento do retângulo) é maior. Isso sugere que durante um maior período de execução ou sob uma maior demanda o impacto nesses dispositivos seria ainda mais crítico.

Os resultados do tráfego de entrada exibidos na figura 26 representam toda requisição

recebida através da rede durante os testes em cada dispositivo. Essa figura nos permite identificar um maior volume de dados requerido pelo serviço de vídeo no Teste_2, bem como uma maior variação entre eles principalmente depois da mediana (2º quartil), indicando alguns picos na comunicação que representam os momentos dessas requisições de vídeo. Sendo assim, os dispositivos apresentaram as seguintes médias: 16,31 KB/s (CP), 59,70 KB/s (FN1) e 24,93 KB/s (FN2). Esses valores representam uma maior taxa de *download* nos dispositivos da *fog*. No Teste_1 o volume de entrada ficou próximo em todos os dispositivos, onde o CP apresentou uma média de 2,99 KB/s, o FN1 de 1,94 KB/s e o FN2 de 1,42 KB/s. Como o *script* executado no Teste_1 simula o acesso dos usuários e a navegação dos mesmos interagindo com a central de mídia através de cliques, é comum termos momentos com maiores requisições do que outros, e por isso uma variação na comunicação capaz de aparacer *outliers* conforme ocorrido com o FN1.

Figura 26 – Tráfego de entrada durante os testes 1 e 2.



Fonte: Própria do autor.

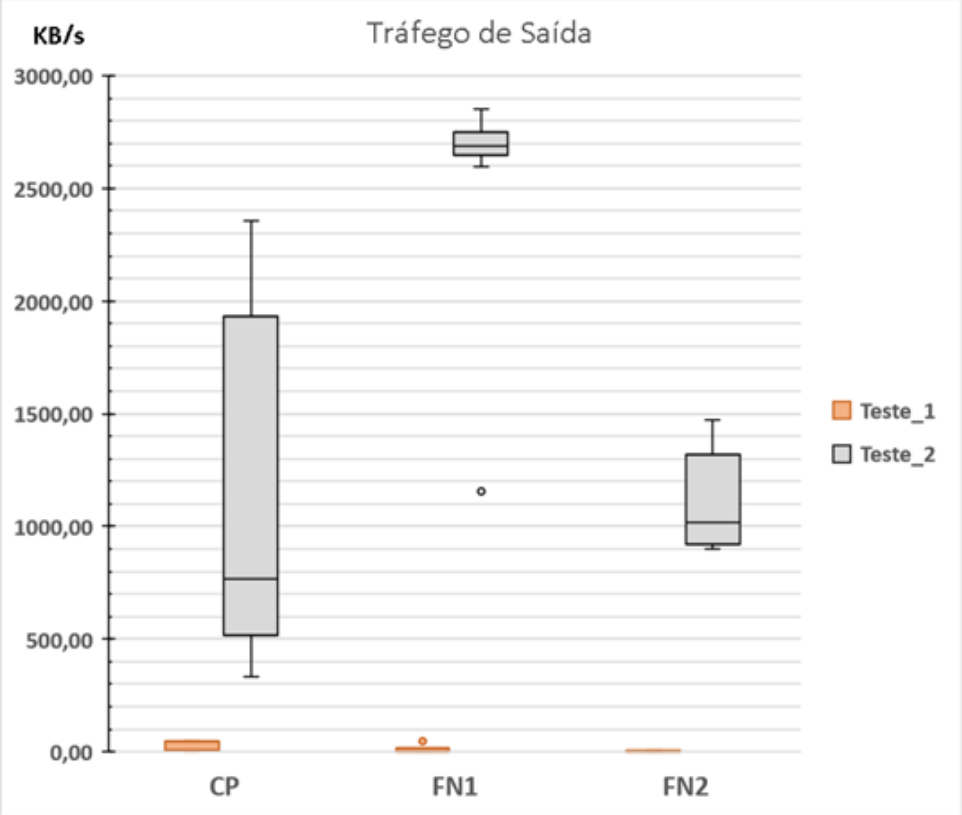
Como no Teste_1 a autenticação para acesso à central de mídia é realizada na nuvem pública pelo outro servidor desta camada, esta etapa do *script* teve um maior impacto no desempenho da AFCV, onde o FN1 e o FN2 fizeram menos requisições devido ao tempo de execução dessa comunicação externa que acaba sendo maior do que no CP que já se encontra na mesma camada.

Devido ao tipo de comunicação usada no acesso à nuvem (WAN) e à AFCV (LAN),

quando há um maior volume de dados trafegados como ocorrido no Teste_2, a conexão do CP varia mais do que no FN1 e FN2. Isso pode ser percebido na figura 26 pela amplitude das caixas, o que significa uma maior imprecisão na comunicação, diferentemente da AFCV que se apresenta mais estável.

Na figura 27, O tráfego de saída confirma as mesmas indicações do tráfego de entrada porém com um volume muito maior de dados devido estarem relacionados à entrega do serviço, e que de fato é bem maior na transmissão de vídeo (Teste_2). As taxas médias de transmissão nesse teste foram de 1082,15 KB/s no CP, 2557,42 KB/s no FN1 e 1117,78 KB/s no FN2. Se olhamos a amplitude dos dados referentes ao dispositivo CP podemos perceber de forma ainda mais clara uma maior instabilidade pela utilização da Internet.

Figura 27 – Tráfego de saída durante os testes 1 e 2.



Fonte: Própria do autor.

No Teste_2, os nós da *fog* apresentaram dados com menor dispersão. O FN1 transferiu mais dados em menos tempo durante esse teste. Para um melhor entendimento, isso significa que durante a transmissão de *streaming* de vídeo por esse dispositivo, os mesmos foram carregados nos clientes (EP1 e EP2) de forma mais rápida, ou seja, os clientes tiveram melhor taxa de *download* e o FN1 melhor taxa de *upload*. Isso explica a redução do volume de dados nesse dispositivo no último minuto de teste, quando já havia entregue mais da metade dos vídeos requisitados, ocasionando o *outlier* existente que simboliza a redução no volume de dados.

O Teste_1 também manteve suas características, com tráfego médio de 21,99 KB/s no CP, 10,81 KB/s no FN1 e 3,67 KB/s no FN2. O que leva ao mesmo problema referente à autenticação comentado na análise do tráfego de entrada. O *outlier* ocorrido no FN1 se deu em resposta às requisições recebidas naquele momento como vimos no tráfego de entrada.

Quanto à escalabilidade aplicada no Teste_1, os resultados demonstram que o acréscimo gradativo de 10 usuários durante os testes não foi suficiente para impactar no aumento do consumo de recursos, considerando que do 5º ao 10º minuto dos testes (quando estava com 50 usuários) houve médias menores do que nos minutos entre 1 e 5 quando possuía menos usuários conectados.

Os resultados obtidos nos experimentos aqui expostos confirmam a hipótese de que com a AFCV e as métricas apresentadas neste trabalho, conseguimos uma maior eficiência no fornecimento de serviços que integram ambientes inteligentes como o de *smart home*, e é possível identificar o consumo de recursos para o gerenciamento dos mesmos e auxílio no fornecimento de QoS e SLA's. Porém, também evidenciam que para demandas bem maiores do que as que foram geradas pelas cargas aqui implementadas, pode haver a necessidade de incluir equipamentos mais robustos na *fog* para o atendimento destes serviços.

No capítulo 6 são apresentadas as conclusões desta dissertação asseguradas por estes resultados, assim como levantamos os desafios futuros a fim de fornecermos um direcionamento dos estudos na área. Também são apresentadas as publicações realizadas e submetidas geradas a partir deste trabalho.

6
Conclusão

Neste capítulo são apresentadas as conclusões relacionadas à esta dissertação, assim como os trabalhos futuros e as publicações realizadas.

6.1 Conclusão da Dissertação

Nesta dissertação de mestrado desenvolvemos e implementamos uma arquitetura de *fog computing* virtualizada (AFCV) (como demonstrado no capítulo 4) para provimento de gerenciamento de recursos, QoS e auxílio para contratos de acordo de nível de serviço (SLA). Como base para a elaboração da hipótese e desenvolvimento da AFCV, no capítulo 3 realizamos um levantamento sobre a conjuntura da *fog computing* através do processo de mapeamento sistemático.

A AFCV projetada possui 3 camadas (nuvem, *fog* e endpoints/IoT) e foi implementada em ambiente real com o uso de ferramentas de virtualização, serviços virtualizados e distribuídos. Isso possibilitou o alcance de diferentes objetivos, tais como, gerenciamento dos recursos e da qualidade dos serviços, fornecimento de parâmetros configuráveis para auxílio no desenvolvimento de contratos de SLA, confiabilidade do serviço, minimização de questões referentes à latência e fornecimento de métricas apropriadas. Também foram geradas cargas com a ferramenta Iperf e o serviço de central de mídia (Plex) que possibilitaram desenvolver os experimentos e realizar as análises de desempenho através das métricas predefinidas.

Diante das análises realizadas no capítulo 5, pudemos destacar o baixo consumo de recursos pelos serviços virtuais implementados na AFCV (sem carga de teste), onde os maiores consumos da arquitetura impactou em 2,45% do consumo de CPU no dispositivo FN1, 2927 MB no consumo de memória pelo dispositivo CP/FCN e 1960 B/s no tráfego de entrada e 1150 B/s no tráfego de saída do dispositivo CP. Destacamos também a superioridade da AFCV comparada à nuvem convencional em relação à eficiência na comunicação, onde a AFCV superou a nuvem

em 52,97% com o protocolo UDP e em 79,72% com o protocolo TCP.

A maior largura de banda conseguida pela AFCV possibilitou melhores taxas de *download* (59,70 KB/s no Fog Node 1 contra 16,31 KB/s da nuvem) e *upload* (2557,42 KB/s no Fog Node 1 contra 1082,15 KB/s da nuvem) na entrega de *streaming* de vídeo entre o cliente e o servidor.Os resultados também mostraram que com as ferramentas implementadas para gerência e controle da AFCV foi possível obter métricas capazes de colaborar para a elaboração de contratos de SLA, contribuindo para a comercialização do serviço de *fog computing*.

O balanceamento de carga se mostrou uma ótima opção para diminuir a sobrecarga nos dispositivos da *fog*, e contribuiu com a qualidade do serviço mesmo nos dispositivos com menos recursos. Isso indica que provedores de menor capacidade e receita podem ser capazes de fornecer este tipo de serviço com qualidade e garantindo aos seus clientes a entrega conforme acordado previamente através de contratos de SLA.

Também abordamos outras diferenças detectadas entre a nuvem e a AFCV, como a falta de controle que temos sobre os nossos dados através da Internet, pois a largura de banda depende de configurações intrínsecas à cada provedor (equipamentos e infraestrutura utilizada). Na AFCV, por ser um ambiente mais restrito diante da proximidade com os dispositivos finais, o ambiente acaba sendo melhor controlado.

Essas análises permitiram comprovar a veracidade da hipótese, de que é possível prover o gerenciamento de recursos, QoS e auxiliar contratos de SLA em ambientes inteligentes (como o de *smart home*), adequando virtualização à estrutura de *fog computing*.

6.2 Trabalhos Futuros

Durante o desenvolvimento desta dissertação, apesar dos esforços empregados em sua elaboração, identificamos algumas questões de melhoria que se resolvidas podem aprimorar o desempenho da AFCV, contribuindo para as pesquisas no âmbito da *fog computing*. A seguir apresentamos essas questões com o objetivo de direcionar trabalhos futuros:

- Pesquisar e implementar aspectos relacionados à segurança, possibilitando a autenticação da aplicação na própria *fog* sem a necessidade de consultar a nuvem conforme foi realizado na AFCV. Desta forma, a arquitetura poderá obter um ganho ainda maior em relação à latência, bem como ser mais segura, pois os dados da autenticação ficarão internamente, sem precisar trafegar pela internet.

- A AFCV permite utilizar outros tipos de balanceamento de carga, sendo assim, realizar pesquisa sobre esses tipos de balanceamento existentes e comparar os mais adequados para um ambiente *fog* seria plausível. Como exemplo, poderia se buscar por balancear o tráfego para o nó da *fog* mais próximo do cliente, com menor congestionamento (JIANG; YAHYA;

ANANTA, 2014), com menor latência, ou dentre outras características que venham a suprir melhor as demandas.

- Distribuir e armazenar os arquivos nos nós da *fog* à medida em que esses forem requisitando, ou seja, os arquivos inicialmente ficariam no FCN que distribuiria a listagem do conteúdo existente para os FN's. Assim que um cliente solicitasse um arquivo dessa lista ao FN em que está conectado, esse buscaria o arquivo no FCN e armazenaria em disco, reduzindo o processo, e consequentemente a latência na próxima requisição.

- Devido às limitações de armazenamento apresentadas pelos nós da *fog* (FN1 e FN2), também se faz necessário a busca por soluções que garantam uma melhor utilização desse recurso. Uma possível solução seria estabelecer que quando algum nó receba uma nova solicitação do cliente e seu disco tenha atingido um limite de armazenamento predefinido, seja excluído de forma automática aquele arquivo que possua mais tempo sem ser utilizado ou tenha recebido menos requisições dos clientes, liberando espaço para que o nó possa copiar o novo arquivo do repositório principal.

6.3 Publicação Realizada

- **A fog computing architecture for security and quality of service.** (QUALIS A2)

  BARRETO, B. N.; SA, A. R. de; RIBEIRO, A. de R. L. A fog computing architecture for security and quality of service. In: GANZHA, M.; MACIASZEK, L.; PAPRZYCKI, M. (Ed.).Position Papers of the 2019 Federated Conference on Computer Science and Information Systems. PTI, 2019. (Annals of Computer Science and Information Systems, v. 19), p. 69–73.Disponível em: <http://dx.doi.org/10.15439/2019F348>

6.4 Publicação Submetida

- **A Virtualized Fog Computing Architecture to Provide Resource Management, QoS and SLA in an Intelligent Environment.** (QUALIS A3)

  BARRETO, B. N.; RIBEIRO, A. de R. L. A Virtualized Fog Computing Architecture to Provide Resource Management, QoS and SLA in an Intelligent Environment. Concurrency and Computation: Practice and Experience (2020)

Referências

ACM. *ACM Digital Library*. 2018. Disponível em: <https://dl.acm.org/>. Acesso em: 04 de out. de 2018. Citado na página 28.

AI, Y.; PENG, M.; ZHANG, K. Edge computing technologies for Internet of Things: a primer. *Digital Communications and Networks*, Elsevier Ltd, v. 4, n. 2, p. 77–86, 2018. ISSN 23528648. Disponível em: <https://doi.org/10.1016/j.dcan.2017.07.001>. Citado 3 vezes nas páginas 14, 20 e 23.

AL-DOGHMAN, F. et al. A review on Fog Computing technology. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, p. 001525–001530, 2016. Disponível em: <http://ieeexplore.ieee.org/document/7844455/>. Citado 6 vezes nas páginas 14, 15, 16, 19, 20 e 22.

ALHAMAD, M.; DILLON, T.; CHANG, E. Conceptual SLA framework for cloud computing. In: *4th IEEE International Conference on Digital Ecosystems and Technologies - Conference Proceedings of IEEE-DEST 2010, DEST 2010*. [S.l.: s.n.], 2010. ISBN 9781424455539. ISSN 2150-4938. Citado 3 vezes nas páginas 24, 44 e 46.

AMADEO, M. et al. On the integration of information centric networking and fog computing for smart home services. *Internet of Things*, Springer International Publishing, Part F2, p. 75–93, 2019. ISSN 21991073. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051751846&doi=10.1007%2F978-3-319-96550-5_4&partnerID=40&md5=3166f4d4faa08e889e20d19be55808cb>. Citado na página 33.

AMADEO, M. et al. A Cloud of Things framework for smart home services based on Information Centric Networking. In: *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*. [S.l.: s.n.], 2017. p. 245–250. Citado 2 vezes nas páginas 32 e 35.

ATZORI, L.; IERA, A.; MORABITO, G. The Internet of Things: A survey. *Computer Networks*, Elsevier B.V., v. 54, n. 15, p. 2787–2805, 2010. ISSN 13891286. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2010.05.010>. Citado 2 vezes nas páginas 14 e 21.

BLAZEMETER. *BlazeMeter, an Enterprise-grade Continuous Testing Platform*. 2019. Disponível em: <https://www.blazemeter.com/>. Acesso em: 24 de out. de 2019. Citado 2 vezes nas páginas 18 e 51.

BONOMI, F. et al. Fog Computing and Its Role in the Internet of Things. p. 13–15, 2012. Citado 4 vezes nas páginas 14, 19, 23 e 24.

BRUNEO, D. et al. An IoT Testbed for the Software Defined City Vision: The #SmartMe Project. In: *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*. [S.l.: s.n.], 2016. p. 1–6. Citado na página 31.

CAPES, P. *Portal .periodicos. CAPES*. 2018. https://www.periodicos.capes.gov.br/ p. Disponível em: <https://www.periodicos.capes.gov.br/>. Citado na página 28.

CARMO, M. S. et al. Towards fog-based slice-defined WLAN infrastructures to cope with future 5G use cases. In: *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*. [S.l.: s.n.], 2017. p. 1–5. Citado 6 vezes nas páginas 16, 17, 32, 35, 38 e 40.

CASTILHO G. U.; KAMIENSKI, C. A. Aplicação de Computação em Névoa na Internet das Coisas para Cidades Inteligentes : da Teoria à Prática. p. 1–14, 2018. Citado na página 23.

CHEN, Y. D.; AZHARI, M. Z.; LEU, J. S. Design and implementation of a power consumption management system for smart home over fog-cloud computing. *IGBSG 2018 - 2018 International Conference on Intelligent Green Building and Smart Grid*, IEEE, p. 1–5, 2018. Citado na página 21.

CLUSTER, I. Internet of T. E. R. Internet of Things 2012 New Horizons. Halifax, UK, 2012. Disponível em: <http://www.internet-of-things-research.eu/pdf/IERC_Cluster_Book_2012_WEB.pdf>. Citado na página 22.

DOCKER. *Empowering App Development for Developers | Docker*. 2018. Disponível em: <https://www.docker.com/>. Acesso em: 20 de out. de 2018. Citado 2 vezes nas páginas 18 e 42.

GITHUBPLEX. *GitHub - plexinc/pms-docker: Plex Media Server Docker repo, for all your PMS docker needs.* 2019. Disponível em: <https://github.com/plexinc/pms-docker>. Acesso em: 4 de fev. de 2019. Citado na página 49.

GOMES, M. et al. Cloud vs Fog: assessment of alternative deployments for a latency-sensitive IoT application. *Procedia Computer Science*, Elsevier B.V., v. 130, p. 488–495, 2018. ISSN 1877-0509. Disponível em: <https://doi.org/10.1016/j.procs.2018.04.059>. Citado na página 23.

HENSHUE, C. G.; HENSHUE, G. L.; RICE, J. C. *Tactile warning panel apparatus and system with smart technology*. 2017. Citado na página 34.

HONG, H.; TSAI, P.; HSU, C. Dynamic module deployment in a fog computing platform. In: *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. [S.l.: s.n.], 2016. p. 1–6. Citado 4 vezes nas páginas 32, 35, 38 e 40.

IEEE. *IEEE Xplore Digital Library*. 2018. Disponível em: <http://ieeexplore.ieee.org/Xplore/home.jsp>. Acesso em: 04 de out. de 2018. Citado na página 28.

INPI. 2018. Disponível em: <https://gru.inpi.gov.br/pePI/jsp/patentes/PatenteSearchBasico.jsp>. Acesso em: 04 de out. de 2018. Citado na página 29.

IPERF. *iPerf - The TCP, UDP and SCTP network bandwidth measurement tool*. 2019. Disponível em: <https://iperf.fr/>. Acesso em: 22 de out. de 2019. Citado 2 vezes nas páginas 18 e 52.

JAIN, R. *The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling*. [S.l.: s.n.], 1991. v. 491. 1–685 p. ISSN 03029743. ISBN 978-0-471-50336-1. Citado 6 vezes nas páginas 36, 38, 54, 55, 59 e 61.

JIANG, J.-R.; YAHYA, W.; ANANTA, M. T. Load balancing and multicasting using the extended dijkstra's algorithm in software defined networking. In: *ICS*. [S.l.: s.n.], 2014. Citado na página 70.

KIM, W.-S.; CHUNG, S.-H. User-Participatory Fog Computing Architecture and Its Management Schemes for Improving Feasibility. *IEEE Access*, v. 6, p. 1–1, 2018. ISSN 2169-3536. Disponível em: <http://ieeexplore.ieee.org/document/8319964/>. Citado na página 20.

LI, L.; LI, S.; ZHAO, S. QoS-Aware scheduling of services-oriented internet of things. *IEEE Transactions on Industrial Informatics*, IEEE, v. 10, n. 2, p. 1497–1507, 2014. ISSN 15513203. Citado 6 vezes nas páginas 15, 16, 17, 24, 36 e 38.

MADAKAM, S.; RAMASWAMY, R. 100 new smart cities (india's smart vision). *2015 5th National Symposium on Information Technology: Towards New Smart World, NSITNSW 2015*, IEEE, p. 1–6, 2015. ISSN 0954-7894. Citado na página 21.

MAHMUD, R.; KOTAGIRI, R.; BUYYA, R. Fog Computing: A Taxonomy, Survey and Future Directions. p. 1–28, 2016. ISSN 14653362. Disponível em: <http://arxiv.org/abs/1611.05539%0Ahttp://dx.doi.org/10.1007/978-981-10-5861-5_5>. Citado 8 vezes nas páginas 14, 15, 19, 20, 21, 24, 36 e 48.

MASIP-BRUIN, X. et al. Managing resources continuity from the edge to the cloud: Architecture and performance. *Future Generation Computer Systems*, v. 79, p. 777–785, 2018. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X17302686>. Citado 3 vezes nas páginas 24, 32 e 35.

MASIP-BRUIN, X. et al. Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems. *IEEE Wireless Communications*, v. 23, n. 5, p. 120–128, 2016. ISSN 1536-1284. Citado 3 vezes nas páginas 16, 31 e 35.

MOURADIAN, C. et al. A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges. *IEEE Communications Surveys and Tutorials*, v. 20, n. 1, p. 416–464, 2018. ISSN 1553877X. Citado 9 vezes nas páginas 15, 16, 17, 24, 32, 36, 37, 43 e 48.

NETO, A. J. et al. Fog-based crime-assistance in smart IoT transportation system. *IEEE Access*, v. 6, p. 11101–11111, 2018. ISSN 21693536. Citado 2 vezes nas páginas 32 e 35.

NGINX. *NGINX | High Performance Load Balancer, Web Server, & Reverse Proxy*. 2019. Disponível em: <https://www.nginx.com/>. Acesso em: 20 de set. de 2019. Citado 2 vezes nas páginas 18 e 46.

NGINX Docs | HTTP Load Balancing. 2020. Disponível em: <https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/#method>. Acesso em: 16 de ago. de 2020. Citado na página 47.

PETERSEN, K. et al. Systematic Mapping Studies in Software Engineering. *12Th International Conference on Evaluation and Assessment in Software Engineering*, v. 17, p. 10, 2008. ISSN 02181940. Disponível em: <http://www.cse.chalmers.se/~feldt/publications/petersen_ease08_sysmap_studies_in_se.pdf>. Citado 4 vezes nas páginas 26, 27, 28 e 30.

PLEX. *Watch Free Movies & TV | Stream Smarter with Plex*. 2019. Disponível em: <https://www.plex.tv/>. Acesso em: 04 de fev. de 2019. Citado 2 vezes nas páginas 18 e 49.

PORTAINER. *Portainer Management, Docker User Interface, Container Software - Auckland, Singapore, San Francisco | Emerging Technology Partners*. 2018. Disponível em: <https://www.portainer.io/>. Acesso em: 16 de nov. de 2018. Citado 2 vezes nas páginas 18 e 47.

PRODAN, R.; OSTERMANN, S. A survey and taxonomy of infrastructure as a service and web hosting cloud providers. *Proceedings - IEEE/ACM International Workshop on Grid Computing*, IEEE, p. 17–25, 2009. ISSN 15505510. Citado 3 vezes nas páginas 15, 16 e 48.

RAY, P. P. A survey on Internet of Things architectures. *Journal of King Saud University - Computer and Information Sciences*, King Saud University, v. 30, n. 3, p. 291–319, 2016. ISSN 22131248. Disponível em: <https://doi.org/10.1016/j.jksuci.2016.10.003>. Citado na página 14.

RED5. *Open Source | Red5 Pro*. 2019. Disponível em: <https://www.red5pro.com/open-source>. Acesso em: 4 de fev. de 2019. Citado na página 49.

REINSEL, D.; GANTZ, J.; RYDNING, J. The Digitization of the World From Edge to Core. n. November, 2018. Disponível em: <https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf>. Citado na página 15.

SANDVINE. *The Global Internet Phenomena Report*. October, 2018. Disponível em: <https://www.sandvine.com/hubfs/downloads/phenomena/2018-phenomena-report.pdf>. Acesso em: 16 de dez. de 2019. Citado na página 49.

SANTOS, J. et al. Anomaly detection for Smart City applications over 5G low power wide area networks. In: *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*. [S.l.: s.n.], 2018. p. 1–9. ISSN 2374-9709. Citado na página 32.

SANTOS, J. et al. Resource provisioning for IoT application services in smart cities. In: *2017 13th International Conference on Network and Service Management (CNSM)*. [S.l.: s.n.], 2017. p. 1–9. ISSN 2165-963X. Citado na página 32.

SCIENCEDIRECT. *ScienceDirect.com | Science, health and medical journals, full text articles and books*. 2018. Disponível em: <https://www.sciencedirect.com>. Acesso em: 04 de out. de 2018. Citado na página 28.

SCOPUS. *Scopus - Document search*. 2018. Disponível em: <http://www.scopus.com>. Acesso em: 04 de out. de 2018. Citado na página 28.

SHARMA, P. K.; CHEN, M. Y.; PARK, J. H. A Software Defined Fog Node Based Distributed Blockchain Cloud Architecture for IoT. *IEEE Access*, v. 6, p. 115–124, 2018. ISSN 21693536. Citado 3 vezes nas páginas 14, 15 e 19.

SILVA, E. L. da; MENEZES, E. M. Metodologia da Pesquisa e Elaboração de Dissertação. 2005. Citado na página 17.

SILVA, I. et al. *Almanaque para Popularização de Ciência da Computação Série 6: Metodologia Científica e Tecnológica*. 2018. Citado 2 vezes nas páginas 26 e 34.

STREAMA. *Create your Home Media Streaming Server for Free*. 2019. Disponível em: <https://medevel.com/streama-create-your-home-media-streaming-server-for-free/>. Acesso em: 4 de fev. de 2019. Citado na página 49.

TECKELMANN, R.; REICH, C.; SULISTIO, A. Mapping of cloud standards to the taxonomy of interoperability in IaaS. *Proceedings - 2011 3rd IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2011*, IEEE, p. 522–526, 2011. Citado 3 vezes nas páginas 15, 16 e 24.

VERMESAN, O.; FRIESS, P.; FURNESS, A. The internet of things 2012 new horizons. *Internet of Things European Research Cluster, 3rd edition of the Cluster Book*, 2012. Citado na página 15.

VOHRA, K.; DAVE, M. Multi-Authority Attribute Based Data Access Control in Fog Computing. *Procedia Computer Science*, Elsevier B.V., v. 132, p. 1449–1457, 2018. ISSN 18770509. Disponível em: <https://doi.org/10.1016/j.procs.2018.05.078>. Citado na página 14.

WAZLAWICK, R. S. Metodologia de Pesquisa para Ciência da Computação. *Decision Support Systems*, v. 38, n. 4, p. 557–573, 2009. ISSN 01679236. Citado na página 38.

WHAT kind of CPU do I need for my Server? | Plex Support. 2019. Disponível em: <https://support.plex.tv/articles/201774043-what-kind-of-cpu-do-i-need-for-my-server/>. Acesso em: 16 de out. de 2019. Citado 2 vezes nas páginas 50 e 63.

WIPO. *WIPO - Search International and National Patent Collections*. 2018. Disponível em: <https://patentscope.wipo.int/search/en/structuredSearch.jsf>. Acesso em: 29 de out. de 2018. Citado 3 vezes nas páginas 29, 33 e 34.

WOWZA. *Wowza Media Systems | Live Video Streaming Solutions*. 2019. Disponível em: <https://www.wowza.com>. Acesso em: 4 de fev. de 2019. Citado na página 49.

YI, S.; LI, C.; LI, Q. A Survey of Fog Computing. *Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata '15*, n. June 2015, p. 37–42, 2015. Disponível em: <http://dl.acm.org/citation.cfm?doid=2757384.2757397>. Citado 7 vezes nas páginas 14, 16, 17, 20, 21, 23 e 24.

Anexos

ANEXO A – Artigo Publicado

Preprints of Position Papers of the Federated Conference on
Computer Science and Information Systems pp. 67–71

A Fog Computing Architecture for Security and Quality of Service

Bruno Nunes Barreto
Federal University of Sergipe
Av. Marechal Rondon, s/n,
Sao Cristovao, Sergipe, Brazil
Email: bnbarreto@gmail.com

Alexandre Rezende de Sa
Federal University of Sergipe
Av. Marechal Rondon, s/n,
Sao Cristovao, Sergipe, Brazil
Email: alexandresa@ufs.br

Admilson de Ribamar Lima Ribeiro
Federal University of Sergipe
Av. Marechal Rondon, s/n,
Sao Cristovao, Sergipe, Brazil
Email: admilson@ufs.br

***Abstract*—The Fog Computing paradigm is an emerging architecture and focuses on optimizing resources for the Internet of Things environment, bringing to the Edge, Cloud's characteristics. The demand generated by the number of possible devices in this network attracts problems related to quality of service, security, among others, attracting researchers from the most diverse areas. In our work, in addition to performing a study on selected works in a mapping process, detecting trends in the use of Fog architectures. The main contribution is presented by a security-based Fog Computing architecture using QoS for scalable environments with Docker containers for orchestration and deployment of security with SDN.**

I. Introduction

THE TECHNOLOGICAL evolution of embedded equipment has enabled virtual communication with certain objects so that we can manage and operate them at a distance through the Internet. With a finality of increase the interactional capacity in systems, a new paradigm called generically Internet of Things (IoT) has been emerging [1].

Through the integration of the most varied technologies, it aims to enable network communication between people, objects and things with different levels of autonomy, extracting and / or providing services and information among themselves or to other devices through the Internet. The IoT architecture can be treated as a physical, virtual or hybrid system, being able to make use of technologies such as Cloud Computing [2], able to overcome the limitations of computing and storage in intelligent devices, besides providing elastic resources to them [3]. According [3], [4] and [5], due to the need to support mobility, geographical distribution, location recognition and low latency demand for some applications, the Cloud meet with some difficulties.

To overcome these difficulties, Cloud features were brought to the edge of the network [6], [7] and [8], thus forming Fog Computing, or simply Fog, which, as a link between IoT and Cloud, induces the extra functionalities required for specific processing of applications, such as filtering and aggregation, before transferring the data to the Cloud [9].

Taking advantage of IoT's capabilities, a wide range of intelligent solutions and applications for the most diverse areas, such as Smart City and Smart Home, have been proposed and increasingly demanded of it, with forecast growth in equipment usage to 50 billion units by 2022, including sensors and actuators [10]. However, this generation advance has been presented in a highly complex way, which has been demanding and moving researchers from the most varied areas of knowledge, besides the need to create environments for the performance analysis of these studies. Some challenges of Fog are listed by [5] and [11], which consider the importance of identifying appropriate techniques and metrics for efficient resource provisioning and management.

In [12], they states that a large number of links and different interactions between edge nodes in IoT makes it a complex and scalable system; therefore, it is difficult to achieve the dynamic requirements of Quality of Service (QoS). How described in [13] and [11] argue that the absence of Service Level Agreement (SLA) management, as well as sustainable metrics, make it difficult to maintain a QoS acceptable in highly dynamic environments. This increase in the number of devices on the network also creates security-related issues, making these endpoints an easy target for malicious people to compromise these devices for use in large-scale attacks.

Thus, our paper aims to present a Fog Computing architecture to provide QoS and security through an orchestrated and virtualized environment, including characteristics such as interoperability and scalability.

The remainder of this paper is structured as follows: Section 2 describes the Fog Computing architeture applied in this study. Next, section 3 presents the implementation issues, followed by related works in the section 4. Section 5 presents a Conclusion.

II. Fog Computing Architecture

According to the paper presented by [14], six criteria considered important for the Fog Computing architecture are: Heterogeneity, QoS Management, Scalability, Mobility, Federation and Interoperability. The architecture we propose next, not only to meet some of these criteria, as well as aspects related to security, providing a consistent, manageable, and secure environment with characteristics that may facilitate the commercialization of services implemented.

This study was financed in part by the Coordenacao de Aperfeicoamento de Pessoal de Nivel Superior - Brasil (CAPES) - Finance Code 001.

These approaches, when contemplated by other articles, are solved individually or in smaller numbers, as we can observe in topic IV (related works). In addition, we are not aware of the use of the K-means algorithm in a Fog Computing.

The proposal of our paper is based on the use of a three-layer architecture similar to that proposed by [4], concomitantly contemplating the six functional blocks of IoT presented by [2] (devices, comunication, services, management, security and application).

As we can see from the Fig. 1, the architecture presents the layers in a well-defined way, where we have the tradicional Cloud at one end, the Fog at the interim layer (composed by Fog nodes) and the edge with the IoT devices. The IoT devices are one of the aforementioned functional blocks, being sensors, actuators, smartphones, among others capable of generating and consuming Fog data.

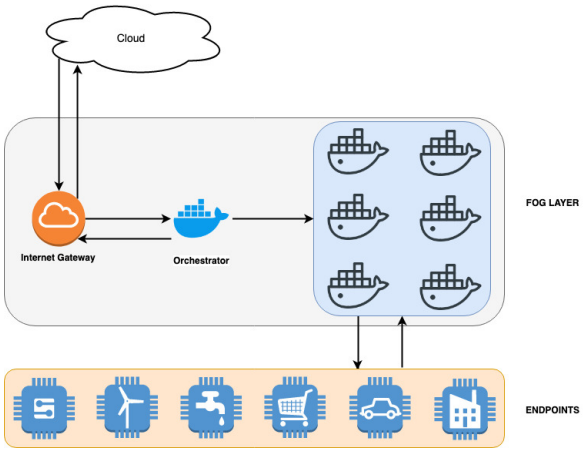


Fig. 1. Fog Archteture proposal.

Interoperability between layers and devices is achieved through the functional block of communication, by means of the data links and their virtualized infrastructure, since much of it is based on Docker containers. Although there are other solutions that promote the use of containers, the Docker offers fault tolerance, service management and deployment capabilities that facilitate the solution delivery process.

Virtualization is a strong trend in the implementation of Fog Computing architectures as we will see in the related works of this paper, making it possible to meet the federation criterion if it becomes a standard used by other service providers, in addition, it makes the architecture scalable through use of a swarm structure, allowing to act in order to deliver the solution continuously orchestrated, attending to the service block.

The last three functional blocks (management, security, and application) are served by another strong feature of this architecture, which is being presented at a time when the threat detection models begin to act directly in Fog layer, allowing the time to decision making is reduced as internal and external threats are identified, thus improving QoS.

This structure will rely on the use of an unsupervised artificial intelligence algorithm capable of learning about anomalies and behavior (DDoS) in a distributed way, which is one of the ten major security flaws in a Fog architecture, according to [15].

As can be seen in the work of [16] and [17], the use of the K-means algorithm presented a very high hit rate compared to other techniques. This algorithm will run in the Cloud (Fig. 2) for training and validation of the samples. Will learn by behavior patterns from open source datasets and then send information to the orchestrator at Fog Computing who will be responsible for generating metrics about the environment as well as resource provisioning computational linked to the models learned in the Cloud.

The orchestrator registers the status of all fog nodes, including the activation and disconnection of nodes, the type of nodes and the IP address of each of them, and is responsible for managing the resources of those nodes that will communicate with the endpoints.

The resources management, among its characteristics, will enable the architecture to simultaneously meet the demands of applications that have or do not have restrictions in real time, prioritizing the guarantee of resources for the most needed or that there is an SLA contract with the client.

The model is combined with the use of SDN (Software Defined Networking) devices since it will be responsible for performing the traffic routing to the endpoints, as well as assisting in the detection, since these data are processed by the Fog node and the cluster, thus providing a better distribution of responsibilities and lower latency among taxpayers. The gateway aims to effect separation and translation between the external and internal networks.

III. Implementation Issues

In order for the environment to achieve the objectives proposed by our architecture, we have a hardware and software structure that will be described as follows:

For anomaly processing solution will be used an Amazon Web Services (AWS) as Cloud Computing Services to find patterns of DDoS attacks. The displayed gateway will be set by a raspberry pi 3 device running the Raspbian Stretch Lite operating system. In order to be orchestrated, 2 physical machines configured with 3.2 Ghz i5 processors and 8 GB ram DDR3 memory will be used, running the linux operating system in the debian 9.9 distribution, as well as the Docker Community edition in version 17.12.1-ce where the portainer management configured, with the portainer/portainer image being available in the Docker hub, chosen for this experiment.

This tool contributes the orchestration of the services in a facilitated way through the use of the webhooks, increasing the practicality in the process of automation of deploy of the final application in the containers that will be destined to solution of SDN.

The area marked in blue in Fig. 2 is responsible for manipulating Fog traffic with IoT devices, applying the rules learned through the cloud and identifying it as malicious or not according to its characteristics. Considering that the entire decision-making process should be automated, this

environment is supported by SDN, responsible for providing the necessary intelligence and automation, creating intelligent routes according to the context.

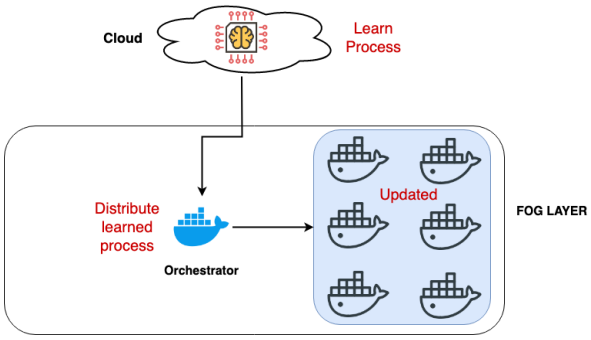


Fig. 2. Fog Security Service.

IV. Related Work

In this session, we discussed the studies considered relevant to our work and commented on the tools, devices, and algorithms most used by them, aiming at a better view of trends and how research in the field of Fog Computing has been developing.

Most of the papers were identified through the systematic mapping process proposed by [18], where we considered the Fog Computing architectures approach that involved both quality of service issues including performance analysis for intelligent environments, as well as questions of security. In this way, we analyze these issues separately in order to facilitate understanding.

A. Related works to architecture in QoS context:

In [19], is present a layered architecture called Fog-to-Cloud (F2C) and compare with an optimized F2C (OF2C) and the traditional Cloud, presenting through simulation and use case in health care the benefits of running services in the different F2C layers. As a result, using the Tareador and Paraver tools, the authors demonstrate an improvement in the task execution speed of 32,05% of the OF2C architecture in relation to the traditional Cloud and poses as a challenge the creation of resource management strategies in different layers of F2C to provide QoS.

A implement through simulation with a framework called Stack4Things, structure based on OpenStack that includes IaaS (Infrastructure as a Service) and PaaS (Platform as a Service) is presented by [20]. They also present a case study of environmental data collection through #SmartME (project to stimulate the creation of a new virtual ecosystem of smart city for the Messina's city). This work indicates anothers types of services that will be provided by Fog, reinforcing the suggestion made by [11] that adapting the Cloud SLA to Fog Computing may be a possible solution for the implementation of this agreement and also as an aid to QoS.

The efficient sharing of client network resources covered by [21], creates network layers configured using SDN and VNF deployed on low-cost common network devices (EX: Raspberry) to approximate wireless and custom services of mobile devices and sensors . As a result, the average cloud delay was approximately 133 ms, versus 12 and 5.3 ms for single board and PC computers, respectively. The environment configured in this work is approximated with the outline of our proposal.

An anomaly detection solution for the smart city application based on Fog, connected to LPWAN and evaluated through algorithms in the testbed of the city of Antwerp is proposed by [22]. The results show that both the Birch cluster and the RC anomaly detection mechanisms can be executed by Fog features. The LPWAN technologies evaluated and validated for the application of air quality were: IEEE 802.11ah, DASH7 and LTE-M.

In [23] study the issue of resource continuity and coordinated Fog and Cloud management and propose the fundamental blocks for system architecture. They demonstrate the benefits of a layer management approach by considering the size and time to search for smart city databases. The authors observed that the smaller the city area the smaller the database size, the lookup time, the lower the number of services to be executed, and thus the lower the interest in these services by the users.

The use of the SDN architecture in a Fog Computing architecture is proposed by [24], focusing on real-time vehicle traffic management, seeking performance enhancement and improved traffic management and QoS in real-time data distribution. In this work, an architecture similar to the one proposed in this paper is used, but its objective is to use it in a vehicular environment.

B. Works related to security issues:

Presented by [25] on his work about Deep Learning on despite the success of traditional Internet cryptographic solutions, factors such as system development flaws, increased attack surfaces, and hacking skills have proven the inevitability of detection mechanisms. Traditional approaches to machine-based attack detection have been successful in the last few decades, but it has already been proven that they have low accuracy and less scalability for detecting cyber attacks on massively distributed nodes such as IoT. The proliferation of deep learning and technological advancement of hardware can pave the way for the detection of the current level of sophistication of cyber attacks in high-end networks. The application of deep networks has already been successful in large areas of data, and this indicates that end-to-end computing may be the ultimate beneficiary of the attack detection approach because a large amount of data produced by IoT devices that deep models learn better than surface algorithms and showed that Deep Learning (DL) models perform well when using unsupervised learning in Zero Day applications, improving model accuracy in invisible and mutant attacks.

In [26] has defined Fog Computing as a new paradigm with many different features of Cloud Computing. Because features are limited, Mobile Edge Computing (MEC) Fog nodes / hosts

are vulnerable to cyber attacks. IDS is a fundamental technique for solving the problem. As the Extreme Learning Machine (ELM) has the characteristics of rapid training speed and good generalization capability, a new light IDS called the extreme selection machine (SS-ELM) is presented. The reason why this new model is proposed is justified because the Fog nodes / MEC hosts do not have the capacity to store extremely large amounts of training data sets. Thus, they are stored, calculated and sampled by the Cloud servers. Then the selected sample is supplied to the Fog / MEC hosts for training. This design can reduce training time and increase detection accuracy. The experimental simulation verifies if the SS-ELM shows good intrusion detection performance in terms of accuracy, training time and receiver operating characteristic (ROC) value.

According to the work of [27] as proposed for IoT applications in which it uses Fog Computing to implement an intrusion detection based on the distributed model. The proposed system consists of two modules: Detection of Fog node attacks and summarization on a Cloud server. In this work the Extreme Learning Machine (ELM) algorithm was used and from it a variant called Online Sequential ELM (OS-ELM) was created to identify the attacks in the inbound traffic of IoT virtual clusters.

In [28] proposed to use the deep learning approach to understand that for the treatment of a large data demand, this algorithm is resilient against metamorphosis attacks with high detection accuracy. In this work it is proposed the use of an LSTM network for detecting distributed cyber attacks in Fog communication for things. The experiments conducted demonstrate the effectiveness and efficiency of deeper models compared to traditional models of machine learning.

As shown in the article of [16], it was proposed a DDoS detection model with K-Means algorithm customization that compared to other works provided a higher rate of detection of anomalies, taking into account factors such as True Positive Rate, False Positive Rate and Recall Rate. In addition, is used the main Open Source Dataset (DARPA, CAIDA, CICIDS), as well as the real-world dataset to proposed benchmark. It forms very high hit rates compared to related jobs.

V. Conclusion

The systematic mapping process used in this paper was extremely important for the direction in the search for the state of the art, resulting in the theoretical basis and the identification of the current conjuncture of Fog computing as a whole reported in this paper through the introduction and related works. This corroborated for a better view of the architectural tendencies, devices and tools used in a Fog environment, such as we also indicate in our work. The virtualization and testbeds, for example, are quite common in the environment in question.

In this paper, we present a Fog Computing architecture capable of providing a consistent, manageable, secure environment with specific characteristics relevant to a Fog and to IoT, such as interoperability, scalability, management, among others. This is due to the fact that we have used a virtualized, orchestrated and intelligent environment, a structure that can facilitate the service delivery process between the existing layers in a secure way. The ability to replicate internal security in an agile way is another important aspect to note.

As future work, this architecture model can be validated through simulations, emulations or even applied in production environments, since in the presented model the SDN was used in the application mode through the software Open vSwitch, however, it is interesting to substitute this model by a professional SDN switch. Taking into consideration that the object of study of our work is Fog and its operation, it was not taken into account the fact of security problems in Cloud Computing, and this issue should be treated in another paper with this focus.

The QoS covered in our work makes it possible the service guarantee, contributing to the business aspect of Fog Computing, which is usually the service level agreement (SLA), negotiated between the service provider and the client, but the commercialization of services involving the Fog still need to be researched.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010. doi: 10.1016/j.comnet.2010.05.010. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2010.05.010

[2] P. P. Ray, "A survey on Internet of Things architectures," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 3, pp. 291–319, 2016. doi: 10.1016/j.jksuci.2016.10.003. [Online]. Available: https://doi.org/10.1016/j.jksuci.2016.10.003

[3] S. Yi, C. Li, and Q. Li, "A Survey of Fog Computing," *Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata '15*, no. June 2015, pp. 37–42, 2015. doi: 10.1145/2757384.2757397. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2757384.2757397

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," pp. 13–15, 2012.

[5] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog Computing: A Taxonomy, Survey and Future Directions," pp. 1–28, 2016. doi: 10.1007/978-981-10-5861-5_5. [Online]. Available: http://arxiv.org/abs/1611.05539%0Ahttp://dx.doi.org/10.1007/978-981-10-5861-5_5

[6] P. K. Sharma, M. Y. Chen, and J. H. Park, "A Software Defined Fog Node Based Distributed Blockchain Cloud Architecture for IoT," *IEEE Access*, vol. 6, pp. 115–124, 2018. doi: 10.1109/ACCESS.2017.2757955

[7] Y. Ai, M. Peng, and K. Zhang, "Edge computing technologies for Internet of Things: a primer," *Digital Communications and Networks*, vol. 4, no. 2, pp. 77–86, 2018. doi: 10.1016/j.dcan.2017.07.001. [Online]. Available: https://doi.org/10.1016/j.dcan.2017.07.001

[8] K. Vohra and M. Dave, "Multi-Authority Attribute Based Data Access Control in Fog Computing," *Procedia Computer Science*, vol. 132, pp. 1449–1457, 2018. doi: 10.1016/j.procs.2018.05.078. [Online]. Available: https://doi.org/10.1016/j.procs.2018.05.078

[9] F. Al-Doghman, Z. Chaczko, A. R. Ajayan, and R. Klempous, "A review on Fog Computing technology," *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 001 525–001 530, 2016. doi: 10.1109/SMC.2016.7844455. [Online]. Available: http://ieeexplore.ieee.org/document/7844455/

[10] K. Yasumoto, H. Yamaguchi, and H. Shigeno, "Survey of Real-time Processing Technologies of IoT Data Streams," *Journal of Information Processing*, vol. 24, no. 2, pp. 195–202, 2016. doi: 10.2197/ipsjjip.24.195

[11] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 1, pp. 416–464, 2018. doi: 10.1109/COMST.2017.2771153

[12] L. Li, S. Li, and S. Zhao, "QoS-Aware scheduling of services-oriented internet of things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497–1507, 2014. doi: 10.1109/TII.2014.2306782

[13] R. Prodan and S. Ostermann, "A survey and taxonomy of infrastructure as a service and web hosting cloud providers," *Proceedings - IEEE/ACM International Workshop on Grid Computing*, pp. 17–25, 2009. doi: 10.1109/GRID.2009.5353074

[14] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 1, pp. 416–464, 2018. doi: 10.1109/COMST.2017.2771153

[15] A. Aljumah and T. A. Ahanger, "Fog computing and security issues: A review," in *2018 7th International Conference on Computers Communications and Control (ICCCC)*, May 2018. doi: 10.1109/ICCCC.2018.8390464 pp. 237–239.

[16] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-supervised k-means ddos detection method using hybrid feature selection algorithm," *IEEE Access*, vol. PP, pp. 1–1, 05 2019. doi: 10.1109/ACCESS.2019.2917532

[17] M. I. W. Pramana, Y. Purwanto, and F. Y. Suratman, "Ddos detection using modified k-means clustering with chain initialization over landmark window," in *2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, Aug 2015. doi: 10.1109/ICCEREC.2015.7337056 pp. 7–11.

[18] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering," *12Th International Conference on Evaluation and Assessment in Software Engineering*, vol. 17, p. 10, 2008. doi: 10.1142/S0218194007003112. [Online]. Available: http://www.cse.chalmers.se/~feldt/publications/petersen_ease08_sysmap_studies_in_se.pdf

[19] X. Masip-Bruin, E. Marï£¡n-Tordera, G. Tashakor, A. Jukan, and G. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 120–128, 2016. doi: 10.1109/MWC.2016.7721750

[20] D. Bruneo, S. Distefano, F. Longo, and G. Merlino, "An IoT Testbed for the Software Defined City Vision: The #SmartMe Project," in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2016. doi: 10.1109/SMARTCOMP.2016.7501678 pp. 1–6.

[21] M. S. Carmo, S. Jardim, A. V. Neto, R. Aguiar, and D. Corujo, "Towards fog-based slice-defined WLAN infrastructures to cope with future 5G use cases," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, 2017. doi: 10.1109/NCA.2017.8171397 pp. 1–5.

[22] J. Santos, P. Leroux, T. Wauters, B. Volckaert, and F. D. Turck, "Anomaly detection for Smart City applications over 5G low power wide area networks," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018. doi: 10.1109/NOMS.2018.8406257. ISSN 2374-9709 pp. 1–9.

[23] X. Masip-Bruin, E. Marin-Tordera, A. Jukan, and G.-J. Ren, "Managing resources continuity from the edge to the cloud: Architecture and performance," *Future Generation Computer Systems*, vol. 79, pp. 777–785, 2018. doi: https://doi.org/10.1016/j.future.2017.09.036. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X17302686

[24] K. S. Sahoo and B. Sahoo, "SDN Architecture on Fog Devices for Realtime Traffic Management : A Case Study SDN architecture on fog devices for realtime traffic management : A case study," no. October, 2017. doi: 10.1007/978-81-322-3592-7

[25] A. Abeshu and N. Chilamkurti, "Deep Learning: The Frontier for Distributed Attack Detection in Fog-To-Things Computing," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 169–175, 2018. doi: 10.1109/MCOM.2018.1700332

[26] X. An, X. Zhou, X. Lü, F. Lin, and L. Yang, "Sample selected extreme learning machine based intrusion detection in fog computing and MEC," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–10, 2018. doi: 10.1155/2018/7472095

[27] S. Prabavathy, K. Sundarakantham, and S. M. Shalinie, "Design of cognitive fog computing for intrusion detection in Internet of Things," *Journal of Communications and Networks*, vol. 20, no. 3, pp. 291–298, 2018. doi: 10.1109/JCN.2018.000041

[28] A. Diro and N. Chilamkurti, "Leveraging LSTM Networks for Attack Detection in Fog-to-Things Communications," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 124–130, 2018. doi: 10.1109/MCOM.2018.1701270