SBRC 2016

XXXIV Simpósio Brasileiro
de Redes de Computadores
e Sistemas Distribuídos

30 de maio a 03 de junho de 2016
Salvador - Bahia - Brasil
www.sbrc2016.ufba.br

Foto Pierre Verger ©Fundação Pierre Verger

LIVRO TEXTO **MINICURSOS**

REALIZAÇÃO

computação UFBA

UFBA 70 ANOS

INSTITUTO FEDERAL BAHIA

SBRC 2016

XXXIV Simpósio Brasileiro
de Redes de Computadores
e Sistemas Distribuídos

30 de maio a 03 de junho de 2016
Salvador - Bahia - Brasil
www.sbrc2016.ufba.br

Livro de Minicursos
SBRC 2016

Editora

Sociedade Brasileira de Computação (SBC)

Organização

Frank Augusto Siqueira (UFSC)
Lau Cheuk Lung (UFSC)
Fabíola Gonçalves Pereira Greve (UFBA)
Allan Edgard Silva Freitas (IFBA)

Realização

Universidade Federal da Bahia (UFBA)
Instituto Federal da Bahia (UFBA)

Promoção

Sociedade Brasileira de Computação (SBC)
Laboratório Nacional de Redes de Computadores (LARC)

Capa: Gilson Rabelo (UFBA)
Produção Editorial: Antonio Augusto Teixeira Ribeiro Coutinho (UEFS)

Cópias Adicionais:
Sociedade Brasileira de Computação (SBC)
Av. Bento Gonçalves, 9500- Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia - CEP 91.509-900 - Porto Alegre - RS
Fone: (51) 3308-6835
E-mail: sbc@sbc.org.br

Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (34: 2016: Salvador, BA).

Minicursos / XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos; organizado por Frank Augusto Siqueira, Lau Cheuk Lung, Fabíola Gonçalves Pereira Greve, Allan Edgard Silva Freitas - Porto Alegre: SBC, 2016

315 p. il. 21 cm.
Vários autores
Inclui bibliografias
ISSN: 2177-4978

1. Redes de Computadores. 2. Sistemas Distribuídos. I. Siqueira, Frank Augusto II. Lung, Lau Cheuk III. Greve, Fabíola Gonçalves Pereira IV. Freitas, Allan Edgard Silva V. Título.

Sociedade Brasileira da Computação

Presidência
Lisandro Zambenedetti Granville (UFRGS), Presidente
Thais Vasconcelos Batista (UFRN),Vice-Presidente

Diretorias
Renata de Matos Galante (UFGRS), Diretora Administrativa
Carlos André Guimarães Ferraz (UFPE), Diretor de Finanças
Antônio Jorge Gomes Abelém (UFPA), Diretor de Eventos e Comissões Especiais
Avelino Francisco Zorzo (PUC RS), Diretor de Educação
José Viterbo Filho (UFF), Diretor de Publicações
Claudia Lage Rebello da Motta (UFRJ), Diretora de Planejamento e Programas Especiais
Marcelo Duduchi Feitosa (CEETEPS), Diretor de Secretarias Regionais
Eliana Almeida (UFAL), Diretora de Divulgação e Marketing

Diretorias Extraordinárias
Roberto da Silva Bigonha (UFMG), Diretor de Relações Profissionais
Ricardo de Oliveira Anido (UNICAMP), Diretor de Competições Científicas
Raimundo José de Araújo Macêdo (UFBA), Diretor de Cooperação com Sociedades Científicas
Sérgio Castelo Branco Soares (UFPE), Diretor de Articulação com Empresas

Contato
Av. Bento Gonçalves, 9500
Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia
91.509-900 – Porto Alegre RS
CNPJ: 29.532.264/0001-78
http://www.sbrc.org.br

Laboratório Nacional de Redes de Computadores (LARC)

Diretora do Conselho Técnico-Científico
Rossana Maria de C. Andrade (UFC)

Vice-Diretor do Conselho Técnico-Científico
Ronaldo Alves Ferreira (UFMS)

Diretor Executivo
Paulo André da Silva Gonçalves (UFPE)

Vice-Diretor Executivo
Elias P. Duarte Jr. (UFPR)

Membros Institucionais
SESU/MEC, INPE/MCT, UFRGS, UFMG, UFPE, UFCG (ex-UFPB Campus Campina Grande), UFRJ, USP, PUC-Rio, UNICAMP, LNCC, IME, UFSC, UTFPR, UFC, UFF, UFSCar, IFCE (CEFET-CE), UFRN, UFES, UFBA, UNIFACS, UECE, UFPR, UFPA, UFAM, UFABC, PUCPR, UFMS, UnB, PUC-RS, UNIRIO, UFS e UFU.

Contato
Universidade Federal de Pernambuco - UFPE
Centro de Informática - CIn
Av. Jornalista Anibal Fernandes, s/n
Cidade Universitária
50.740-560 - Recife - PE
http://www.larc.org.br

Organização do SBRC 2016

Coordenadores Gerais
Fabíola Gonçalves Pereira Greve (UFBA)
Allan Edgard Silva Freitas (IFBA)
Romildo Martins Bezerra (IFBA) - In Memoriam

Coordenadores do Comitê de Programa
Antonio Marinho Pilla Barcellos (UFRGS)
Antonio Alfredo Ferreira Loureiro (UFMG)

Coordenador de Palestras e Tutoriais
Francisco Vilar Brasileiro (UFCG)

Coordenador de Painéis e Debates
Carlos André Guimarães Ferraz (UFPE)

Coordenadores de Minicursos
Lau Cheuck Lung (UFSC)
Frank Siqueira (UFSC)

Coordenadora de Workshops
Michele Nogueira Lima (UFPR)

Coordenador do Salão de Ferramentas
Daniel Macêdo Batista (USP)

Comitê de Organização Local
Adolfo Duran (UFBA)
Allan Freitas (IFBA)
Antonio Augusto Teixeira Ribeiro Coutinho (UEFS)
Cátia Khouri (UESB)
Fabíola Greve (UFBA)
Flávia Maristela Nascimento (IFBA)
Gustavo Bittencourt (UFBA)
Ítalo Valcy (UFBA)
Jauberth Abijaude (UESC)
Manoel Marques Neto (IFBA)
Marco Antônio Ramos (UESB)
Marcos Camada (IF BAIANO)
Marcos Ennes Barreto (UFBA)
Maycon Leone (UFBA)
Rafael Reale (IFBA)
Renato Novais (IFBA)
Ricardo Rios (UFBA)
Romildo Bezerra (IFBA) - In Memoriam
Sandro Andrade (IFBA)
Vinícius Petrucci (UFBA)

Comitê Consultivo
Jussara Almeida (UFMG), Coordenadora
Elias Procópio Duarte Jr. (UFPR)
José Rezende (UFRJ)
Jacir Luiz Bordim (UnB)
Rafael Timóteo de Sousa Júnior (UnB)
William Ferreira Giozza (UnB)
Carlos André Guimarães Ferraz (UFPE)
José Augusto Suruagy Monteiro (UFPE)

Mensagem dos Coordenadores Gerais

Sejam bem-vindos ao 34o Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) e à nossa terra, Salvador da Bahia.

O desafio de organizar um evento do porte, escopo e excelência do SBRC é substancial. Agradecemos à comunidade pela confiança e privilégio em poder contribuir para a mais tradicional e relevante conferência científica nacional na sua área e uma das mais concorridas da Ciência da Computação na América Latina. No contexto desafiador em que o país se encontra nesse ano de 2016, é ainda mais importante reforçar o papel primordial do SBRC como fórum de discussão, encontros e divulgação dos melhores trabalhos produzidos pela comunidade de pesquisa nacional.

A programação do SBRC 2016 está diversificada, abrangente e tem excelente qualidade; nesse sentido, a contribuição da comunidade foi fundamental para a valorização do evento e o fortalecimento da Ciência e Tecnologia no nosso país. A programação engloba 27 sessões técnicas com apresentação de 81 artigos científicos completos, selecionados por meio de um rigoroso trabalho de revisão, 3 palestras proferidas por pesquisadores internacionalmente renomados, e 3 painéis de discussões e debates, todos em temas atuais, como Computação em Nuvem, Internet da Coisas, Cidades Inteligentes e Experimentação. São oferecidos 6 minicursos, voltados à formação e atualização dos participantes em temas de ponta, e apresentados 2 tutoriais, de forma a aproximar o público dos avanços tecnológicos.

Adicionalmente, há a exposição de 8 ferramentas no Salão de Ferramentas e realização de 8 workshops, em paralelo ao SBRC, com foco em temas de pesquisa e desenvolvimento específicos e emergentes. Nesta edição, através do prêmio "Destaque SBRC", também homenageamos uma personalidade nas áreas de Redes de Computadores e Sistemas Distribuídos, pela sua importante contribuição para o fomento da pesquisa e estruturação de uma sólida comunidade científica no Brasil.

A excelência das atividades programadas nesta edição é reflexo da competência e empenho dos seus respectivos coordenadores. Um agradecimento muito especial a Marinho Barcellos (UFRGS), Antonio Alfredo Ferreira Loureiro (UFMG), Francisco Vilar Brasileiro (UFCG), Carlos André Guimarães Ferraz (UFPE), Lau Cheuck Lung (UFSC), Frank Siqueira (UFSC), Michele Nogueira Lima (UFPR) e Daniel Macêdo Batista (USP). Exaltamos ainda o trabalho voluntário, intenso, atencioso e contínuo, realizado pelos colegas do Comitê de Organização Local e através deles agradecemos o apoio das instituições parceiras: Universidade Estadual de Feira de Santana (UEFS), Universidade Estadual do Sudoeste da Bahia (UESB), Universidade Estadual de Santa Cruz (UESC) e Instituto Federal Baiano (IFBaiano).

Agradecemos às diretorias da SBC e do LARC, promotores do SBRC, pela confiança depositada em nós, e pelo competente apoio organizacional prestado pela equipe administrativa da SBC. Somos também gratos aos integrantes do Comitê Consultivo do SBRC e à coordenação da Comissão Especial de Redes de Computadores e Sistemas Distribuídos da SBC, pelos aconselhamentos e pelo suporte financeiro inicial prestado à organização do SBRC 2016. Gostaríamos ainda de agradecer aos patrocinadores do simpósio: ao Comitê Gestor da Internet no Brasil (CGI.br), aos órgãos de fomento, CNPq, CAPES, e ACM Sigcomm, aos nossos apoiadores e às empresas patrocinadoras por valorizarem e reconhecerem o SBRC como um evento importante para o fomento à pesquisa e inovação. Nosso agradecimento especial às nossas instituições, especialmente, ao Departamento de Ciência da Computação, ao Instituto de Matemática e à Reitoria da UFBA, e ao Departamento de Computação, ao Campus de

Salvador e à Reitoria do IFBA, pelo indispensável suporte para a realização do SBRC.

Por fim, sensibilizados com a perda precoce do nosso amigo e colega Romildo Martins da Silva Bezerra (IFBA), organizador in memoriam dessa edição, gostaríamos de deixar o nosso testemunho da sua grandeza, energia e exemplo de profissional e pesquisador.

Em nome do Comitê Organizador do SBRC 2016, desejamos a todos uma semana agradável em Salvador, rica em discussões e encontros.

Fabíola Greve (UFBA) e Allan Freitas (IFBA)
Coordenadores Gerais do SBRC 2016

Mensagem dos Coordenadores de Minicursos

Os minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2016) constituem uma importante oportunidade para atualização de conhecimentos da comunidade científica e para complementação da formação dos estudantes das áreas de Informática e Engenharias. Através dos minicursos do SBRC, assuntos relevantes e atuais, geralmente não contemplados nas disciplinas de cursos de graduação, são abordados de forma didática e em maior profundidade que em outras atividades do simpósio.

Os livros-texto dos minicursos do SBRC, publicados a cada edição do simpósio, vêm sendo amplamente utilizados como fonte bibliográfica para pesquisadores e educadores, constituindo um importante mecanismo de difusão do conhecimento. O livro-texto dos minicursos é composto por capítulos elaborados pelos autores de cada um dos minicursos apresentados no simpósio, abordando com profundidade o assunto tratado no minicurso.

Nesta edição do SBRC foram recebidas 14 propostas de minicursos, das quais seis foram selecionadas para apresentação durante o simpósio. Acreditamos que a alta qualidade e a diversidade das propostas selecionadas serão apreciadas não somente pelos participantes do simpósio, mas também por todos que tiverem acesso a esse material.

O Comitê de Avaliação dos Minicursos do SBRC 2016 foi composto por 20 pesquisadores com forte atuação na área de conhecimento abordada pelo simpósio, que realizaram uma criteriosa seleção dos trabalhos submetidos. Cada proposta de minicurso foi analisada por pelo menos três avaliadores, que contribuíram com enriquecedores comentários que possibilitaram o aperfeiçoamento dos textos produzidos pelos autores dos minicursos. Expressamos nossos sinceros agradecimentos a todos os membros do comitê por terem realizado voluntariamente este trabalho com extrema competência e dedicação.

A qualidade desta obra se deve, fundamentalmente, ao esforço de cada um dos autores dos minicursos. Somos imensamente gratos a eles pelas horas de trabalho despendidas na elaboração da proposta e do material de instrução.

Agradecemos também ao Comitê de Organização do SBRC 2016, em especial aos Coordenadores Gerais do SBRC 2016, Fabíola Greve, Allan Freitas e Romildo Bezerra (*in memoriam*), por todo o suporte e orientação fornecidos ao longo do processo de seleção de minicursos e de elaboração deste livro.

Finalmente, desejamos a todos os participantes do SBRC 2016 que façam ótimo proveito do conhecimento que lhes está sendo proporcionado por meio dos minicursos e das demais atividades do simpósio, e que desfrutem das inúmeras belezas de Salvador e da hospitalidade do povo baiano!

Frank Siqueira e Lau Cheuk Lung (UFSC)
Coordenadores de Minicursos do SBRC 2016

Comitê de Avaliação

- Aldelir Fernando Luiz (IF Catarinense)
- Aldri dos Santos (UFPR)
- Alex Borges Vieira (UFJF)
- Ana Paula Couto da Silva (UFMG)
- Antonio Rocha (IC/UFF)
- Artur Ziviani (LNCC)
- Carlos Alberto Vieira Campos (UNIRIO)
- Daniel Batista (IME/USP)
- Eduardo Alchieri (UnB)
- Eduardo Souto (UFAM)
- Flavia Delicato (UFRJ)
- Frank Siqueira (UFSC)
- Gustavo Figueiredo (UFBA)
- Lasaro Camargos (UFU)
- Luciana Rech (UFSC)
- Michele Nogueira (UFPR)
- Miguel Correia (IST, Universidade de Lisboa)
- Miguel Elias Mitre Campista (UFRJ)
- Stenio Fernandes (UFPE)

Sumário

Capítulo 1

Internet das Coisas: da Teoria à Prática

Bruno P. Santos, Lucas A. M. Silva, Clayson S. F. S. Celes, João B. Borges Neto, Bruna S. Peres, Marcos Augusto M. Vieira, Luiz Filipe M. Vieira, Olga N. Goussevskaia e Antonio A. F. Loureiro

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, MG, Brasil

{bruno.ps, lams, claysonceles, joaoborges, bperes, mmvieira, lfvieira, olga, loureiro}@dcc.ufmg.br

Abstract

The proliferation of smart objects with capability of sensing, processing and communication has grown in recent years. In this scenario, the Internet of Things (IoT) connects these objects to the Internet and provides communication with users and devices. IoT enables a huge amount of new applications, with which academics and industries can benefit, such as smart cities, health care and automation. On the other hand, there are several social, theoretical and practical issues we need to address. To answer these issues, we need to overcome some challenges like dealing with objects' constraints (e.g., processing, memory and energy supply), limited bandwidth and hardware dimension. Thus, we need to explore new communication paradigms, protocols including questions about IP addressing and adaptations to interoperate with the Internet, hardware architecture and software design. Besides that, IoT applications need to deal with the process of collecting, storing, processing and extracting knowledge from data obtained from the smart objects. The goal of this chapter is to describe the state-of-the-art of IoT by discussing theoretical and practical questions.

Resumo

A proliferação de objetos inteligentes com capacidade de sensoriamento, processamento e comunicação tem aumentado nos últimos anos. Neste cenário, a Internet das Coisas (Internet of Things (IoT)) conecta estes objetos à Internet e promove a comunicação entre usuários e dispositivos. A IoT possibilita uma grande quantidade de novas aplicações, as quais tanto a academia quanto a industria podem se beneficiar, tais como cidades inteligentes, saúde e automação de ambientes. Por outro lado, existem diversos desafios que devemos enfrentar no âmbito social, teórico e prático. Para responder a essas questões, precisamos vencer alguns desafios como, por exemplo, restrições dos objetos inteligentes (processamento, memória e fonte de alimentação), largura de banda limitada e dimensão do hardware. Deste modo, devemos explorar novos paradigmas de comunicação, protocolos incluindo questões sobre o endereçamento IP e adaptações para interoperar com a Internet, arquitetura de hardware e projeto de software. Além disso, aplicações de IoT precisam tratar questões como coletar, armazenar, processar e extrair conhecimento de dados obtidos dos objetos inteligentes. O objetivo deste capítulo é descrever o estado-da-arte de IoT discutindo questões teóricas e práticas.

1.1. Introdução

A Internet das Coisas (do inglês *Internet of Things (IoT)*) emergiu dos avanços de várias áreas como sistemas embarcados, microeletrônica, comunicação e sensoriamento. De fato, a IoT tem recebido bastante atenção tanto da academia quanto da indústria, devido ao seu potencial de uso nas mais diversas áreas das atividades humanas. Este capítulo aborda a Internet das Coisas através de uma perspectiva teórica e prática. O conteúdo aqui abordado explora a estrutura, organização, desafios e aplicações da IoT. Nesta seção, serão conceituadas a IoT e os objetos inteligentes. Além disso, são apresentadas a perspectiva histórica da Internet das Coisas e as motivações que levam aos interesses, expectativas e pesquisas na área. Logo em seguida, são introduzidos os blocos básicos de construção da IoT. Para iniciar a discussão, será levantada a seguinte questão: o que é a Internet das Coisas?

A Internet das Coisas, em poucas palavras, nada mais é que uma extensão da Internet atual, que proporciona aos objetos do dia-a-dia (quaisquer que sejam), mas com capacidade computacional e de comunicação, se conectarem à Internet. A conexão com a rede mundial de computadores viabilizará, primeiro, controlar remotamente os objetos e, segundo, permitir que os próprios objetos sejam acessados como provedores de serviços. Estas novas habilidades, dos objetos comuns, geram um grande número de oportunidades tanto no âmbito acadêmico quanto no industrial. Todavia, estas possibilidades apresentam riscos e acarretam amplos desafios técnicos e sociais.

A IoT tem alterado aos poucos o conceito de redes de computadores, neste sentido, é possível notar a evolução do conceito ao longo do tempo como mostrado a seguir. Para Tanenbaum [Tanenbaum 2002], "Rede de Computadores é um conjunto de computadores autônomos interconectados por uma única tecnologia". Entende-se que tal tecnologia de conexão pode ser de diferentes tipos (fios de cobre, fibra ótica, ondas eletromagnéticas ou outras). Em 2011, Peterson definiu em [Peterson and Davie 2011] que a principal característica das Redes de Computadores é a sua generalidade, isto é, elas são construídas

sobre dispositivos de propósito geral e não são otimizadas para fins específicos tais como as redes de telefonia e TV. Já em [Kurose and Ross 2012], os autores argumentam que o termo "Redes de Computadores" começa a soar um tanto envelhecido devido à grande quantidade de equipamentos e tecnologias não tradicionais que são usadas na Internet.

Os objetos inteligentes, definidos mais adiante, possuem papel fundamental na evolução acima mencionada. Isto porque os objetos possuem capacidade de comunicação e processamento aliados a sensores, os quais transformam a utilidade destes objetos. Atualmente, não só computadores convencionais estão conectados à grande rede, como também uma grande heterogeneidade de equipamentos tais como TVs, *Laptops*, automóveis, *smartphones*, consoles de jogos, *webcams* e a lista aumenta a cada dia. Neste novo cenário, a pluralidade é crescente e previsões indicam que mais de 40 bilhões de dispositivos estarão conectados até 2020 [Forbes 2014]. Usando os recursos deses objetos será possível detectar seu contexto, controlá-lo, viabilizar troca de informações uns com os outros, acessar serviços da Internet e interagir com pessoas. Concomitantemente, uma gama de novas possibilidades de aplicações surgem (ex: cidades inteligentes (*Smart Cities*), saúde (*Healthcare*), casas inteligentes (*Smart Home*)) e desafios emergem (regulamentações, segurança, padronizações). É importante notar que um dos elementos cruciais para o sucesso da IoT encontra-se na padronização das tecnologias. Isto permitirá que a heterogeneidade de dispositivos conectados à Internet cresça, tornando a IoT uma realidade. Também é essencial frisar que nos últimos meses e nos próximos anos serão vivenciados os principais momentos da IoT, no que tange as definições dos blocos básicos de construção da IoT.

Parafraseado os autores [Mattern and Floerkemeier 2010], ao utilizar a palavra "*Internet*"no termo "*Internet of Things*"como acima mencionado, pode-se fazer uma analogia com a Web nos dias de hoje, em que brevemente as "coisas"terão habilidades de comunição umas com as outras, proverão e usarão serviços, proverão dados e poderão reagir a eventos. Outra analogia, agora mais técnica, é que IoT vista como uma pilha de protocolos utilizados nos objetos inteligentes.

Na IoT, eventualmente, a unidade básica de *hardware* apresentará ao menos uma das seguintes características [Ruiz et al. 2004, Loureiro et al. 2003]: i) unidade(s) de processamento; ii) unidade(s) de memória; iii) unidade(s) de comunicação e; iv) unidade(s) de sensor(es) ou atuador(es). Aos dispositivos com essas qualidades é dado o nome de *objetos inteligentes* (*Smart Objects*). Os objetos, ao estabelecerem comunicação com outros dispositivos, manifestam o conceito de estarem em rede, como discutido anteriormente.

1.1.1. Perspectiva histórica da IoT

Kevin Ashton comentou, em junho de 2009 [Ashton 2009], que o termo *Internet of Things* foi primeiro utilizando em seu trabalho intitulado "*I made at Procter & Gamble*" em 1999. Na época, a IoT era associada ao uso da tecnologia RFID. Contudo, o termo ainda não era foco de grande número de pesquisas como pode ser visto na Figura 1.1. Por volta de 2005, o termo bastante procurado (tanto pela academia quando indústria) e que apresenta relação com a IoT foi Redes de Sensores Sem Fio (RSSF) (do inglês *Wireless Sensor Networks* – WSN). Estas redes trazem avanços na automação residencial e industrial [Kelly et al. 2013, Da Xu et al. 2014], bem como técnicas para explorar as

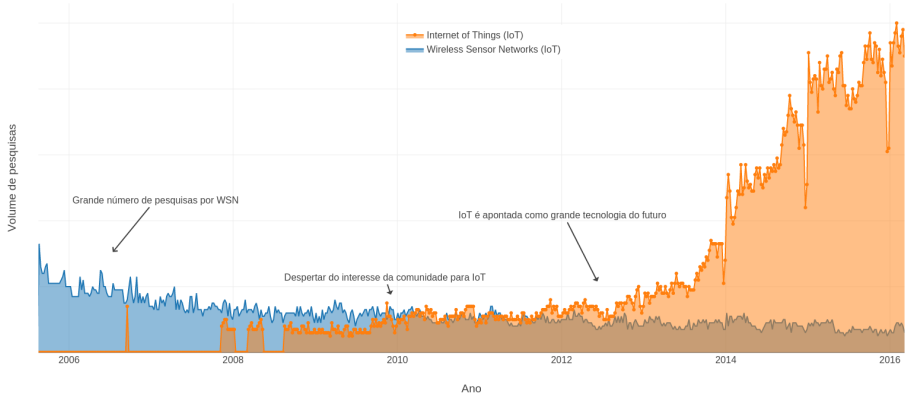


Figura 1.1. Volume de pesquisas no Google sobre *Wireless Sensor Networks* e *Internet of Things*

diferentes limitações dos dispositivos (e.g., memória e energia), escalabilidade e robustez da rede [Loureiro et al. 2003]. Nos anos seguintes (entre 2008 e 2010), o termo Internet das Coisas ganhou popularidade rapidamente. Isto se deve ao amadurecimento das RSSFs e ao crescimento das expectativas sobre a IoT. A Figura 1.1 mostra que em 2010, as buscas para IoT dispararam chegando a ultrapassar as pesquisas sobre RSSFs.

Figura 1.2. Tecnologias emergentes

A IoT foi identificada como uma tecnologia emergente em 2012 por especialistas da área [Gartner 2015]. A Figura 1.2 apresenta uma maneira de representar o surgimento, adoção, maturidade e impacto de diversas tecnologias chamada de *Hype Cycle*, ou "Ciclo de Interesse". O pesquisador Clarke , pesquisador do MIT, propôs a curva dos dois elefantes, que

Em 2012, foi previsto que a IoT levaria entre cinco e dez anos para ser adotada pelo mercado e, hoje, é vivenciado o maior pico de expectativas sobre a tecnologia no âmbito acadêmico e industrial. Também pode-se notar o surgimento das primeiras plataformas de IoT (discutidas mais adiante) que têm gerado uma grande expectativa de seu uso. Estes fatos certamente servem de incentivo para despertar a curiosidade do leitor para a área, bem como indica o motivo do interesse da comunidade científica e industrial para a IoT.

1.1.2. Motivação para manutenção da evolução da IoT

Ao conectar objetos com diferentes recursos a uma rede, potencializa-se o surgimento de novas aplicações. Neste sentido, conectar esses objetos à Internet significa criar a Internet das Coisas. Na IoT, os objetos podem prover comunicação entre usuários, dispositivos. Com isto emerge uma nova gama de aplicações, tais como coleta de dados de pacientes e monitoramento de idosos, sensoriamento de ambientes de difícil acesso e inóspitos, entre outras [Sundmaeker et al. 2010].

Neste cenário, a possibilidade de novas aplicações é crescente, mas temos novos desafios de conectar à Internet objetos com restrições de processamento, memória, comunicação e energia [Loureiro et al. 2003]. Naturalmente, nesse caso os objetos são heterogêneos, isto é, divergem em implementação, recursos e qualidade. Algumas das questões teóricas quanto práticas que surgem são, por exemplo, prover endereçamento aos dispositivos, encontrar rotas de boa vazão e que usem parcimoniosamente os recursos limitados dos objetos. Deste modo, fica evidente a necessidade da adaptação dos protocolos existentes. Além disso, sabe-se que os paradigmas de comunicação e roteamento nas redes de objetos inteligentes podem não seguir os mesmos padrões de uma rede como a Internet [Chaouchi 2013].

Novos desafios surgem enquanto são criadas novas aplicações para IoT. Os dados providos pelos objetos agora podem apresentar imperfeições (calibragem do sensor), inconsistências (fora de ordem, *outliers*) e serem de diferentes tipos (gerados por pessoas, sensores físicos, fusão de dados). Assim, as aplicações e algoritmos devem ser capazes de lidar com esses desafios sobre os dados. Outro exemplo diz respeito ao nível de confiança sobre os dados obtidos dos dispositivos da IoT e como/onde pode-se empregar esses dados em determinados cenários. Deste modo, os desafios impostos por essas novas aplicações devem ser explorados e soluções devem ser propostas para que a IoT contemplem as expectativas em um futuro próximo como previsto na Figura 1.2.

Alguns autores apontam que a IoT será a nova revolução da tecnologia da informação [Ashton 2009, Forbes 2014, Wang et al. 2015]. Sendo assim, a IoT possivelmente não deve ser entendida como um fim, mas sim um meio de alcançar algo maior como a computação ubíqua.

1.1.3. Blocos Básicos de Construção da IoT

A IoT pode ser vista como a combinação de diversas tecnologias, as quais são complementares no sentido de viabilizar a integração dos objetos no ambiente físico ao mundo virtual. A Figura 1.3 apresenta os blocos básicos de construção da IoT sendo eles:

Identificação: é um dos blocos mais importantes, visto que é primordial identificar os objetos unicamente para conectá-los à Internet. Tecnologias como RFID, NFC (*Near Field Communication*) e endereçamento IP podem ser empregados para identificar os objetos.

Sensores/Atuadores: sesnsores coletam informações sobre o contexto onde os objetos se encontam e, em seguida, armazenam/encaminham esses dados para *data warehouse*, *clouds* ou centros de armazenamento. Atuadores podem manipular o ambiente ou reagir de acordo com os dados lidos.

Comunicação: diz respeito às diversas técnicas usadas para conectar objetos inteligen-

tes. Também desempenha papel importante no consumo de energia dos objetos sendo, portanto, um fator crítico. Algumas das tecnologias usadas são WiFi, Bluetooth, IEEE 802.15.4 e RFID.

Computação: inclui a unidade de processamento como, por exemplo, micro-controladores, processadores e FPGAs, responsáveis por executar algoritmos locais nos objetos inteligentes.

Serviços: a IoT pode prover diversas classes de serviços, dentre elas, destacam-se os *Serviços de Identificação*, responsáveis por mapear Entidades Físicas (EF) (de interesse do usuário) em Entidades Virtuais (EV) como, por exemplo, a temperatura de um local físico em seu valor, coordenadas geográficas do sensor e instante da coleta; *Serviços de Agregação de Dados* que coletam e sumarizam dados homogêneos/heterogêneos obtidos dos objetos inteligentes; *Serviços de Colaboração e Inteligência* que agem sobre os serviços de agregação de dados para tomar decisões e reagir de modo adequado a um determinado cenário; e *Serviços de Ubiquidade* que visam prover serviços de colaboração e inteligência em qualquer momento e qualquer lugar em que eles sejam necessários.

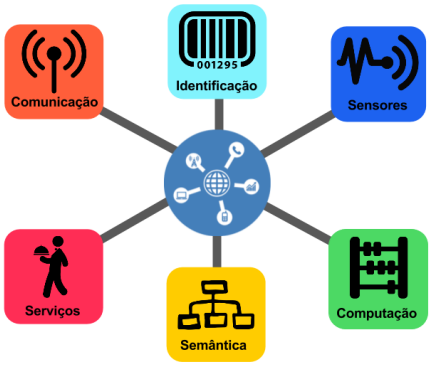


Figura 1.3. Blocos básicos da IoT

Semântica: refere-se à habilidade de extração de conhecimento dos objetos na IoT. Trata da descoberta de conhecimento e uso eficiente dos recursos existentes na IoT, a partir dos dados existentes, com o objetivo de prover determinado serviço. Para tanto, podem ser usadas diversas técnicas como *Resource Description Framework (RDF)*, *Web Ontology Language (OWL)* e *Efficient XML Interchange (EXI)*.

1.1.4. Objetivos do minicurso

Este capítulo tem por objetivo apresentar ao leitor uma visão geral da Internet das Coisas – IoT, o que inclui diversos aspectos das Tecnologias da Informação e Comunicação (TICs). Para isso, apresentaremos o significado e funcionalidade dos principais blocos relacionados à IoT.

Após a breve apresentação dos blocos da IoT, ficará evidente o quão ampla é a área. Em seguida, serão discutidos os seguintes blocos básicos: **Identificação**, **Comunicação** e **Serviços/Semântica**. A discussão tratará não apenas cada bloco individualmente, mas também como se relaciona com os demais. Esses blocos foram escolhidos por capturarem a essência da IoT e possibilitarem ao leitor condições de explorar com maior facilidade outros blocos da IoT.

1.1.5. Estrutura do capítulo

Este capítulo é dividido em seis seções, sendo esta a primeira. A Seção 1.2 aborda pontos sobre os dispositivos e as tecnologias de comunicação para IoT. A Seção 1.3 dis-

cute sobre os softwares que orquestram a operação da IoT, pontuando a identificação única dos dispositivos com IPv6, bem como modelos de conectividade, protocolos de roteamento e aplicação para IoT, e ambientes de desenvolvimento. A Seção 1.4 apresenta duas práticas para consolidar o conteúdo visto até então. Essas práticas serão o elo para o conteúdo da Seção 1.5, a qual aborda o gerenciamento e análise de dados que permeiam os blocos básicos de semântica e serviços na IoT. Finalmente, a Seção 1.6 apresenta as considerações finais e destaca os pontos que não foram cobertos no capítulo, apresentando referências para que o leitor possa complementar seus estudos.

1.2. Dispositivos e Tecnologias de Comunicação

Esta seção aborda em mais detalhes a arquitetura básica dos dispositivos inteligentes e as tecnologias de comunicação, principalmente as soluções de comunicação sem fio que tendem a se popularizar no ambiente de IoT.

1.2.1. Arquiteturas básica dos dispositivos

A arquitetura básica dos objetos inteligentes é composta por quatro unidades: processamento/memória, comunicação, energia e sensores/atuadores. A Figura 1.4 apresenta uma visão geral da arquitetura de um dispositivo e a interligação entre seus componentes, os quais são descritos a seguir:

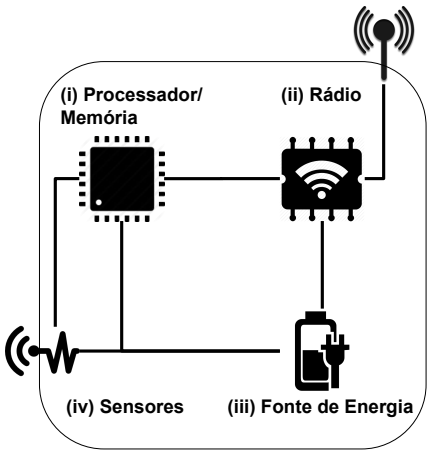


Figura 1.4. Arquitetura dos dispositivos

(i) Unidade(s) de processamento/-memória: composta de uma memória interna para armazenamento de dados e programas, um microcontrolador e um conversor analógico-digital para receber sinais dos sensores. As CPUs utilizadas nesses dispositivos são, em geral, as mesmas utilizadas em sistemas embarcados e comumente não apresentam alto poder computacional. Frequentemente existe uma memória externa do tipo *flash*, que serve como memória secundária, por exemplo, para manter um "*log*" de dados. As características desejáveis para estas unidades são consumo reduzido de energia e ocupar o menor espaço possível.

(ii) Unidade(s) de comunicação: consiste de pelo menos um canal de comunicação com ou sem fio, sendo mais comum o meio sem fio. Neste último caso, a maioria das plataformas usam rádio de baixo custo e baixa potência. Como consequência, a comunicação é de curto alcance e apresentam perdas frequentes. Esta unidade básica será objeto de estudo mais detalhado na próxima seção.

(iii) Fonte de energia: responsável por fornecer energia aos componentes do objeto inteligente. Normalmente, a fonte de energia consiste de uma bateria (recarregável ou não) e um conversor AC-DC e tem a função de alimentar os componentes. Entretanto, existem

outras fontes de alimentação como energia elétrica, solar e mesmo a captura de energia do ambiente através de técnicas de conversão (e.g., energia mecânica em energia elétrica), conhecidas como *energy harvesting*.

(iv) Unidade(s) de sensor(es)/atuador(es): realizam o monitoramento do ambiente no qual o objeto se encontra. Os sensores capturam valores de grandezas físicas como temperatura, umidade, pressão e presença. Atualmente, existem literalmente centenas de sensores diferentes que são capazes de capturar essas grandezas. Atuadores, como o nome indica, são dispositivos que produzem alguma ação, atendendo a comandos que podem ser manuais, elétricos ou mecânicos.

1.2.2. Tecnologias de comunicação

Esta seção trata das principais tecnologias de comunicação utilizadas em IoT, identificando as características mais relevantes de cada um delas.

Ethernet. O padrão Ethernet (IEEE 802.3) foi oficializado em 1983 pelo IEEE e está presente em grande parte das redes locais com fio existentes atualmente. Sua popularidade se deve à simplicidade, facilidade de adaptação, manutenção e custo. Atualmente, existem dois tipos de cabos: par trançado e fibra óptica, que oferecem taxas de comunicação diferentes. Os cabos de par trançado podem atingir taxas de até 1 Gbps (categoria 5), limitados a 100 m (para distâncias maiores é necessário o uso de repetidores). Os cabos de fibra óptica alcançam taxas de 10 Gbps, limitados a 2000 m [Tanenbaum 2011]. O uso do padrão Ethernet é sugerido para dispositivos fixos, i.e., sem mobilidade, o que pode ser inadequado para essas aplicações.

Wi-Fi. A tecnologia Wi-Fi é uma solução de comunicação sem fio bastante popular, pois está presente nos mais diversos lugares, fazendo parte do cotidiano de casas, escritórios, indústrias, lojas comerciais e até espaços públicos das cidades. O padrão IEEE 802.11 (Wi-Fi[1]) define um conjunto de padrões de transmissão e codificação. Desde o seu lançamento em 1997, já foram propostas novas versões do padrão IEEE 802.11 e, atualmente, a versão IEEE 802.11ac prevê taxas de comunicação de 600 Mbps ou 1300 Mbps.

O Wi-Fi foi desenvolvido como uma alternativa ao padrão cabeado Ethernet, com pouca preocupação com dispositivos que possuem consumo energético limitado, como é o caso das aplicações para IoT. Assim, não se espera que muitos dispositivos utilizados em IoT adotem o padrão Wi-Fi como principal protocolo de comunicação. Contudo, o Wi-Fi possui algumas vantagens, como alcance de conexão e vazão, o que o torna adequado para navegação na Internet em dispositivos móveis, como *smartphones* e *tablets*. A principal desvantagem do Wi-Fi é o maior consumo de energia, quando comparado com outras tecnologias de comunicação sem fio.

ZigBee. O padrão ZigBee é baseado na especificação do protocolo IEEE 802.15.4 para a camada de enlace. As suas principais características são a baixa vazão, reduzido consumo energético e baixo custo. O ZigBee opera na frequência 2.4 GHz (faixa ISM), mas é capaz de operar em outras duas frequências, 868 MHz e 915 MHz. Essa tecnologia pode

[1]Por um abuso de linguagem, chamamos o padrão IEEE 802.11 de Wi-Fi, que na verdade representa a "Aliança Wi-Fi" (http://www.wi-fi.org/) responsável por certificar a conformidade de produtos que seguem a norma IEEE 802.11.

alcançar uma taxa máxima de 250 kbps, mas na prática temos taxas inferiores . O ZigBee também permite que os dispositivos entrem em modo *sleep* por longos intervalos de tempo para economizar energia e, assim, estendendo a vida útil do dispositivo.

O padrão ZigBee é mantido pela *ZigBee Alliance*, organização que é responsável por gerir o protocolo e garantir a interoperabilidade entre dispositivos. O padrão ZigBee pode ser usado com o protocolo IP (incluindo o IPv6) e também utilizando a topologia em malha (*Mesh*[2]). O ZigBee já é adotado em aplicações residenciais e comerciais e sua utilização em IoT depende de um *gateway*, dispositivo responsável por permitir a comunicação entre dispositivos que usam ZigBee e os que não usam.

Bluetooth Low Energy. O Bluetooth é um protocolo de comunicação proposto pela Ericsson para substituir a comunicação serial RS-232 [3]. Atualmente, o *Bluetooth Special Interest Group* é responsável por criar, testar e manter essa tecnologia. Além disso, Bluetooth é uma das principais tecnologias de rede sem fio para PANs – *Personal Area Networks*, que é utilizada em *smartphones*, *headsets*, PCs e outros dispositivos. O Bluetooth se divide em dois grupos: Bluetooth clássico que por sua vez se divide em *Basic Rate/Enhanced Data Rate* (BR/EDR), que são as versões 2.0 ou anteriores e o *Bluetooth High Speed* (HS), versão 3.0; e o *Bluetooth Low Energy* (BLE), versão 4.0 ou superior. As versões mais antigas do Bluetooth, focadas em aumentar a taxa de comunicação, tornou o protocolo mais complexo e, por consequência, não otimizado para dispositivos com limitações energéticas. Ao contrário das versões anteriores, o BLE possui especificação voltada para baixo consumo de energia, permitindo dispositivos que usam baterias do tamanho de moedas (*coin cell batteries*).

Atualmente, o BLE possui três versões: 4.0, 4.1 e 4.2, lançadas em 2010, 2013 e 2014, respectivamente. BLE 4.0 e 4.1 possuem um máximo de mensagem (*Maximum Transmit Unit* – MTU) de 27 bytes, diferentemente da mais nova versão (BLE 4.2) que possui 251 bytes. Outra diferença entre as versões é o tipo de topologia de rede que cada versão pode criar. Na versão 4.0, apenas a topologia estrela é disponível, ou seja, cada dispositivo pode atuar exclusivamente como *master* ou como *slave*. A partir da versão 4.1, um dispositivo é capaz de atuar como *master* ou *slave* simultaneamente, permitindo a criação de topologias em malha. Recentemente foi proposta uma camada de adaptação para dispositivos BLE, similar ao padrão 6LoWPAN, chamada de 6LoBTLE. A especificação do 6LoBTLE pode ser consultada na RFC 7668[4].

3G/4G. Os padrões de telefonia celular 3G/4G também podem ser aplicados à IoT. Projetos que precisam alcançar grandes distâncias podem aproveitar as redes de telefonia celular 3G/4G. Por outro lado, o consumo energético da tecnologia 3G/4G é alto em comparação a outras tecnologias. Porém, a sua utilização em locais afastados e baixa mobilidade podem compensar esse gasto. No Brasil, as frequências utilizadas para o 3G são 1900 MHz e 2100 MHz (UMTS), enquanto o padrão 4G (LTE) utiliza a frequência de 2500 MHz. A taxa de comunicação alcançada no padrão 3G é de 1 Mbps e no padrão 4G

[2] http://www.zigbee.org/zigbee-for-developers/network-specifications/

[3] https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-fact-or-fiction

[4] https://tools.ietf.org/html/rfc7668

10 Mbps [5].

LoRaWAN is a Low Power Wide Area Network (LPWAN) specification intended for wireless battery operated Things in regional, national or global network. LoRaWAN target key requirements of internet of things such as secure bi-directional communication, mobility and localization services.

LoRaWan. A especificação LoRaWAN (*Long Range Wide Area Network*) foi projetada para criar redes de longa distância, numa escala regional, nacional ou global, formada por dispositivos operados por bateria e com capacidade de comunicação sem fio. A especificação LoRaWan trata de requisitos presentes na IoT como comunicação segura e bi-direcional, mobilidade e tratamento de serviços de localização. Além disso, o padrão oferece suporte a IPv6, adaptação ao 6LoWPAN e funciona sobre a topologia estrela [6]. O fator atrativo do LoRAWAN é o seu baixo custo e a quantidade de empresas de *hardware* que estão o adotando. A taxa de comunicação alcança valores entre 300 bps a 50 kbps. O consumo de energia na LoRaWan é considerado pequena, o que permite aos dispositivos se manterem ativos por longos períodos. A LoRaWANs utiliza a frequência ISM sub-GHz fazendo com que as ondas eletromagnéticas penetrem grandes estruturas e superfícies, a distâncias de 2 km a 5 km em meio urbano e 45 km no meio rural. Os valores de frequência mais usadas pelo LoRaWan são: 109 MHz, 433 MHz, 866 MHz e 915 MHz. O MTU adotado pelo padrão LoRaWAN é de 256 bytes[7].

Sigfox. O SigFox[8] utiliza a tecnologia *Ultra Narrow Band* (UNB), projetada para lidar com pequenas taxas de transferência de dados. Essa tecnologia ainda é bastante recente e já possui bastante aceitação, chegando a dezenas de milhares de dispositivos espalhados pela Europa e América do Norte. A SigFox atua como uma operadora para IoT, com suporte a uma série de dispositivos. A principal função é abstrair dificuldades de conexão e prover uma API para que os usuários implementem sistemas IoT com maior facilidade. O raio de cobertura oficialmente relatada, em zonas urbanas, está entre 3 km e 10 km e em zonas rurais entre 30 km a 50 km. A taxa de comunicação varia entre 10 bps e 1000 bps e o MTU utilizado é de 96 bytes. O SigFox possui baixo consumo de energia e opera na faixa de 900 MHz.

A Tabela 1.1 resume as tecnologias de comunicação apresentadas nesta seção. As principais características de cada uma são elencadas, o que permite compará-las. Em particular, destaca-se a grande variedade de possibilidades para conectar dispositivos. Portanto, é preciso ponderar acerca das características das tecnologias e finalidade do dispositivo para escolher a melhor forma de conectá-lo.

1.2.3. Perspectivas e desafios sobre os objetos inteligentes

A evolução nas tecnologias de hardware utilizadas em RFID, RSSF, e, consequentemente, na IoT é impressionante quando olhamos apenas a última década. Os dispositivos estão cada vez menores e possuem mais recursos. Pode-se esperar ainda que essa evolução continue e, no futuro, possivelmente veremos outras tecnologias de hardware

[5] https://www.opensensors.io/connectivity
[6] http://www.semtech.com/wireless-rf/lora/LoRa-FAQs.pdf
[7] https://www.lora-alliance.org
[8] http://makers.sigfox.com

Protocolo	Alcance	Frequência	Taxa	IPv6	Topologia
Ethernet	100/2000 m	N/A	10 Gbps	Sim	Variada
Wi-Fi	50 m	2.4/5 GHz	1300 Mbps	Sim	Estrela
BLE	80 m	2.4 GHz	1 Mbps	Sim*	Estrela/Mesh
ZigBee	100 m	915 MHz/2.4 GHz	250 kbps	Sim	Estrela/Mesh
3G/4G	35/200 km	1900/2100/2500 MHz	1/10 Mbps	Sim	Estrela
SigFox	10/50 km	868/902 MHz	10–1000 bps	–	–
LoraWan	2/5 km	Sub-GHz	0.3-50 kbps	Sim	Estrela

Tabela 1.1. Comparação entre as tecnologias de comunicação

empregadas na IoT, diferentes das de hoje. Atualmente, temos dispositivos inteligentes como sistemas-em-um-chip (*system on a chip* – SoC) [Vasseur and Dunkels 2010], que claramente é uma evolução quando analisamos os últimos 10 a 15 anos. Esta seção trata das perspectivas e desafios para a IoT. A seguir, serão abordados dois pontos centrais. No primeiro, a questão da fonte de energia e o segundo as futuras tecnologias de hardware para dispositivo IoT.

1.2.3.1. Bateria e Colheita de Energia (*Energy Harvesting*)

Como mostrado na Figura 1.4, os dispositivos da IoT requerem uma fonte de energia. Atualmente, os objetos inteligentes são alimentados, em geral, por baterias, muitas vezes não recarregáveis. No entanto, existem outras fontes de alimentação como energia elétrica e solar. As baterias (recarregáveis ou não) são as fontes de alimentação mais empregadas nos dispositivos de IoT, embora não sejam as mais adequadas para a tarefa. Isto porque, em geral, os dispositivos estão em locais de difícil acesso (e.g., embutidos em outros dispositivos) ou simplesmente não é desejável manipulá-los fisicamente para substituir as baterias. Assim, tanto o hardware quando o software devem ser projetados para estender ao máximo a vida útil desses dispositivos [RERUM 2015].

Uma possível estratégia para mitigar o problema da energia é fazer uso da técnica de colheita de energia (do inglês *Energy Harvesting*) [Ramos et al. 2012]. A estratégia consiste em transformar energia de fontes externas ao dispositivo como, por exemplo, solar, térmica, eólica e cinética em energia elétrica e armazená-la em uma bateria recarregável. A energia colhida (geralmente em pequenas quantidades) é armazenada e deve ser utilizada para suprir as necessidades dos objetos como comunicação e processamento [Liu et al. 2013]. Contudo, a colheita de energia também traz ao menos um outro desafio que discutiremos a seguir, o qual pode ser o ponto de partida para novas pesquisas.

Caso os dispositivos tenham a capacidade de colher energia do ambiente onde estão inseridos, surgem diversas questões a serem tratadas. Por exemplo, como programar as atividades que o dispositivo deve executar considerando o orçamento de energia (*energy budget*)? Em outras palavras, como gastar a energia em função das atividades que devem ser feitas? Para exemplificar, imagine a situação na qual o dispositivo deve realizar tarefas durante 24 horas/dia. No entanto, somente quando há luz do sol é possível

captar energia do ambiente e o dispositivo não está acessível fisicamente. Além disso, sabe-se que a carga da bateria não consegue fornecer energia para 24 horas de operação. Sendo assim, as atividades do dispositivo devem ser realizadas de modo intermitente, ou seja, alterando entre realizar os comandos requeridos e entrar em modo de espera ou de baixa potência (*sleep mode*). Portanto, a atividade deve ser realizada de modo inteligente dado a demanda de atividades e orçamento de energia residual e adquirida do ambiente. Técnicas mais sofisticadas podem ainda adaptar a computação a ser feita em função da energia disponível ou de expectativas de se ter energia no futuro, em função do que foi possível captar de energia do passado. Nesse caso, isso envolve técnicas de predição.

1.2.3.2. Tecnologias de hardware

Considerando a discussão anterior, fica claro que os objetos inteligentes e tecnologias de comunicação apresentam diferentes características e limitações. À medida que os dispositivos de IoT diminuem, por um lado as limitações tendem a aumentar devido ao espaço reduzido, mas por outro podemos ter avanços tecnológicos que mitiguem essas restrições. Por exemplo, anos atrás foi prevista a possibilidade de existirem dispositivos em escala nanométrica, o que já é uma realidade com os *microelectromechanical systems (MEMS)* [Angell et al. 1983]. Já hoje, existem propostas de utilizarmos *Claytronics* e *Programmable Matter* [Goldstein et al. 2005], que consistem em combinar robôs de escala nanométrica ("*claytronic atoms*" ou "*catoms*") para formar outras máquinas. Os *catoms* eventualmente serão capazes de se comunicarem, moverem e conectarem com outros *catoms* para apresentar diferentes formas e utilidades. Uma outra perspectiva futura quanto ao hardware diz respeito aos SoCs, onde um circuito integrado possui todos os componentes de um elemento computacional. Para o caso dos objetos inteligentes, esse sistema terá rádio, microcontrolador e sensores/atuadores. Entretanto, atualmente isso é um problema, pois o rádio requer um arranjo sofisticado para ser posto em um único chip [Vasseur and Dunkels 2010].

Há poucos anos atrás, era possível apontar o fator custo como limitante para adoção e desenvolvimento de objetos inteligentes. Entretanto, hoje é possível encontrar soluções de IoT disponíveis no mercado de baixo custo. Para esse segmento, é possível afirmar que o custo do hardware já é acessível, se analisarmos o preço de produtos como o Raspberry Pi, Arduino e similares que permitem desde a prototipagem até a produção final de soluções de IoT a baixo custo. Por exemplo, é possível encontrar o Raspberry Pi ao custo de US$ 35.

1.3. Softwares e Ambientes de Desenvolvimentos para IoT

Esta seção aborda assuntos referentes à camada de software que orquestra a lógica de operação dos objetos inteligentes. O bloco básico de identificação e os mecanismos de comunicação são os pontos centrais da seção. Além disso, serão descritos os sistemas operacionais para IoT, bem como os principais ambientes de desenvolvimento. O tópico, a seguir, apresentará alguns argumentos e requisitos para softwares utilizados na IoT. Os desdobramentos dos pontos levantados serão objetos de análise ao longo da seção.

1.3.1. Software para rede de objetos inteligentes

RSSF foi objeto de grande análise por pesquisadores tanto da academia quanto da indústria nos últimos anos como exposto na Seção 1.1. A interconexão entre as RSSF e a Internet foi uma evolução natural. Entretanto, o uso dos protocolos da Internet (arquitetura TCP/IP), sem modificações, é impraticável nos dispositivos com recursos limitados, os quais são comuns na IoT. Para alcançar as demandas da IoT por escalabilidade e robustez, a experiência em RSSF mostra que são necessários algoritmos e protocolos especializados como, por exemplo, protocolos que viabilizem processamento ao longo da rede (*in-network processing*) [Santos et al. 2015b, Fasolo et al. 2007] para mitigar as restrições impostas pelos dispositivos e prover escalabilidade e eficiência. Por outro lado, a arquitetura fim-a-fim da Internet não foi planejada para essas exigências (e.g., dezenas de milhares de dispositivos em uma única sub-rede e limitações de energia). Portanto, tal arquitetura necessita de ajustes para comportar essas novas demandas.

As RSSF e sua evolução, a IoT, cresceram em um ambiente com maior liberdade de desenvolvimento do que a Internet como, por exemplo, a questão de compatibilidade. Diante desta autonomia diversas ideias surgiram tanto do ponto de vista de software quanto de hardware. Entretanto, muitas dessas ideias não avançaram, pois não eram completamente interoperáveis com a arquitetura TCP/IP da Internet. Essas novas ideias, que não empregam a arquitetura da Internet, foram obrigadas a usar um *gateway* para trocar informações entre os objetos inteligentes e elementos na Internet. A implementação de um *gateway* é, em geral, complexa e o seu gerenciamento é complicado, pois além de realizarem funções de tradução, devem tratar da semântica de serviços para a camada de aplicação. A complexidade destas funções torna o *gateway* um gargalo para a IoT em dois sentidos. No primeiro, toda informação de e para a rede de objetos inteligentes transita através do *gateway*. No segundo, a lógica de operação da Internet diz que a inteligência do sistema deve ficar nas pontas [Saltzer et al. 1984], porém com o emprego dos *gateways* a inteligência da IoT fica no meio da rede, o que contradiz com os princípios da arquitetura da Internet. Para tratar desses problemas a *Internet Engineering Task Force (IETF)* criou dois grupos para gerenciar, padronizar e levantar os requisitos para as redes de baixa potência (*Low-Power and Lossy Networks (LLN)*) relacionadas a seguir:

6LoWPAN: o *IPv6 in Low-Power Wireless Personal Area Networks Working Group* ficou responsável por padronizar o *Internet Protocol version 6* (IPv6) para redes que fazem uso de rádios sobre o padrão IEEE 802.15.4 que, por sua vez, especifica as regras das camada mais baixas (enlace e física) para redes sem fio pessoais de baixas potência de transmissão.

RoLL: o *Routing over Low-Power and Lossy Links Working Group* ficou responsável por padronizar o protocolo de roteamento que utilizará o IPv6 em dispositivos com limitações de recursos. O grupo já definiu o protocolo: *IPv6 Routing Protocol for Low-Power and Lossy Networks* (RPL). Esse protocolo representa o estado-da-arte em roteamento para IoT, o qual constrói uma topologia robusta para comunicação na Internet das Coisas. O RPL será objeto de discussão mais adiante.

1.3.2. Arquitetura para IoT

Para conectar bilhões de objetos inteligentes à Internet, deve-se ter uma arquitetura flexível. Na literatura, temos uma variedade de propostas de arquiteturas sofisticadas, que se baseiam nas necessidades da academia e indústria [Al-Fuqaha et al. 2015]. O modelo básico de arquitetura apresenta três camadas, como ilustrado na Figura 1.5. A primeira camada é a de objetos inteligentes ou camada de percepção. Esta camada representa os objetos físicos, os quais utilizam sensores para coletarem e processarem informações. Na camada de rede, as abstrações das tecnologias de comunicação, serviços de gerenciamento, roteamento e identificação devem ser realizados. Logo acima, encontra-se a camada de aplicação, a qual é responsável por prover serviços para os clientes. Por exemplo, uma aplicação solicita medições de temperatura e umidade para clientes que requisitam estas informações.

1.3.3. IP para IoT

Figura 1.5. Arquitetura para IoT

O protocolo IPv4 foi o padrão utilizado para endereçar os dispositivos em rede e criar a "cola" da Internet, i.e., para estar conectado à Internet era necessário ter o protocolo IP. No entanto, não se imaginou que a Internet cresceria e poderia ter dezenas de milhares de pontos finais em uma única sub-rede, tal como agora é previsto para a IoT (veja Seção 1.1). O crescimento da rede mundial de computadores levou ao esgotamento de endereços IPv4 disponíveis. Isto mostrou que o IPv4 não era escalável o suficiente para atender a demanda da IoT.

O IPv6 é uma abordagem mais eficaz para solucionar a escassez de endereços IPv4. Os 32 bits alocados originalmente para o protocolo IPv4 foram expandidos para 128 bits, aumentando imensamente a quantidade de dispositivos endereçáveis na Internet. Na IoT, os elementos da rede são endereçados unicamente usando o IPv6 e, geralmente, têm o objetivo de enviar pequenas quantidades de dados obtidos pelos dispositivos. Contudo, o IPv6 tem um tamanho de pacote maior que o tamanho do quadro dos protocolos usados pelos dispositivos na IoT (o pacote IPv6 é transmitido dentro da área de dados do quadro do protocolo de acesso ao meio). Por exemplo, o padrão IEEE 802.15.4, usado para acesso ao meio físico de comunicação, limita os pacotes a 128 bytes. Para resolver esse problema, a IETF criou o 6LoWPAN [Kushalnagar et al. 2007].

6LoWPAN[9] é uma camada de adaptação primariamente desenvolvida para o padrão IEEE 802.15.4. A principal ideia é realizar a compressão de pacotes IPv6 (ID-6lowpan-hc[10]), permitindo a dispositivos com baixo poder computacional o uso do IPv6. A compressão do 6LoWPAN é possível através da utilização de informações de protocolos presentes em outras camadas. Por exemplo, o 6LoWPAN pode utilizar parte do

[9] https://tools.ietf.org/html/rfc4919
[10] https://tools.ietf.org/html/draft-ietf-6lowpan-hc-15

endereço MAC do dispositivo para atribuir um endereço IPv6 para o objeto inteligente. Desta forma, o 6LoWPAN requer menos bits que o IPv6 para realizar o endereçamento.

O pesquisador Adam Dunkels[11] realizou uma série de contribuições na área da Internet das Coisas. Três de suas principais contribuições são as implementações da pilha TCP/IP para dispositivos de baixa potência. Estas implementações são conhecidas como *low weight IP* (lwIP), o micro IP (μIP) e o sistema operacional para IoT Contiki.

A lwIP é uma implementação reduzida da pilha TCP/IP para sistemas embarcados[12]. A lwIP possui mais recursos do que a pilha μIP, discutida a seguir. Porém, a lwIP pode prover uma vazão maior. Atualmente esta pilha de protocolos é mantida por desenvolvedores espalhados pelo mundo[13]. A lwIP é utilizada por vários fabricantes de sistemas embarcados como, por exemplo, Xilinx e Altera. lwIP conta com os seguintes protocolos: IP, ICMP, UDP, TCP, IGMP, ARP, PPPoS, PPPoE. As aplicações incluídas são: servidor HTTP, cliente DHCP, cliente DNS, ping, cliente SMTP e outros.

O μIP (Micro IP) é uma das menores pilhas TCP/IP existentes[14]. Desenvolvida para pequenos microcontroladores (processadores de 8 ou 16 bits), onde o tamanho do código e memória RAM disponível são escassos, μIP precisa de no máximo 5 kilobytes de espaço de código e de poucas centenas de bytes de RAM. Atualmente, o μIP foi portado para vários sistemas[15] e integra o Contiki.

1.3.4. Modelos de conectividade em redes de objetos inteligentes

Nesta seção, serão abordados os modelos de conectividade para uma rede IPv6 de objetos inteligentes. Os modelos de conectividade serão classificados como um espectro que varia de uma rede de objetos inteligentes autônoma sem conexão com a Internet até a "autêntica" *Internet of Things*, na qual os objetos possuem acesso à Internet efetivamente. As redes de objetos inteligentes serão parte de nossas vidas nos próximos anos. Por isso, entender as bases da IoT no que tange seus paradigmas de comunicação e modelos de conectividade é de grande importância, pois estes dois quesitos serão as chaves para construir novas aplicações para a IoT. A Figura 1.6 destaca três modelos de conectividade de uma rede de objetos inteligentes. A seguir, serão discutidos cada um dos modelos de conectividade: **Rede autônoma de objetos inteligentes:** neste modelo a rede de objetos não possui conexão com a Internet "pública". Existem diversos casos de uso para este modelo como, por exemplo, em uma rede de automação industrial (e.g., usina nuclear). Pode-se requerer uma rede de objetos conectados entre si, porém completamente inacessível externamente, ou seja, o uso da rede é estritamente interno da corporação. A Figura 1.6 (I) apresenta uma abordam esquemática do modelo.

Uma questão que pode ser levantada aqui é: "Neste modelo de conectividade é necessário que os objetos inteligentes sejam endereçados com IP?" Para responder a esta questão é preciso refletir sobre a experiência passada com o IP em redes convencionais. O IP apresenta diversas características que indicam um sim como resposta. Por exemplo,

[11] http://www.dunkels.com/adam/
[12] http://www.ece.ualberta.ca/~cmpe401/docs/lwip.pdf
[13] http://savannah.nongnu.org/projects/lwip/
[14] https://github.com/adamdunkels/uip
[15] http://www.dunkels.com/adam/software.html

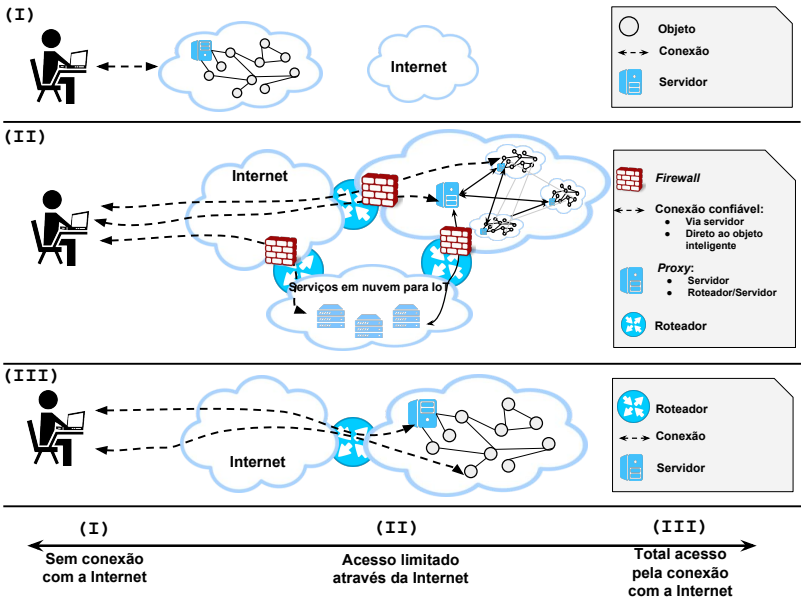


Figura 1.6. Modelos de conectividade dos objetos inteligentes. (I) Rede autônoma na qual os objetos inteligentes não possuem conexão com a Internet; (II) Rede de objetos inteligentes limitada, pois o acesso aos dispositivos é restrito; (III) IoT "autêntica" na qual os objetos estão conectados à Internet

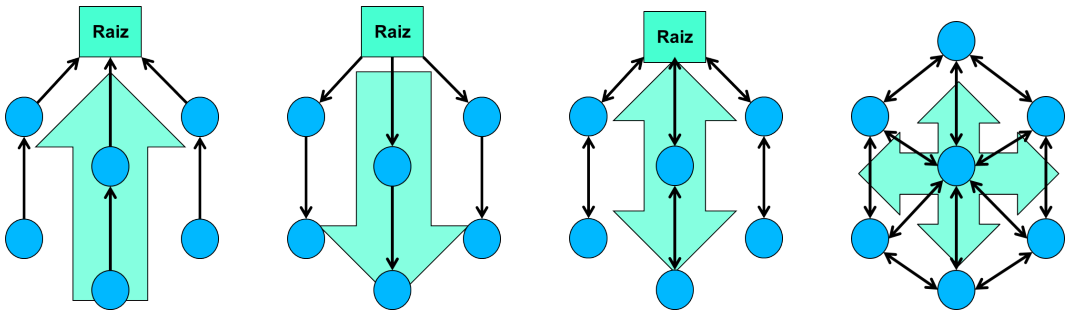
a arquitetura TCP/IP é interoperável, no sentido de que ela opera sobre as camadas de enlace e física com diferentes características. Outro argumento a favor do sim, é que o IP é versátil e evolutivo, pois a arquitetura é baseada no princípio fim-a-fim, em que a inteligência da rede (aplicações) residem nos pontos finais, enquanto a rede é mantida simples (somente encaminhando pacotes). Isto permitiu a evolução contínua do protocolo ao longo do tempo. Tendo dito isto, é válido alegar que ao usar IP neste modelo, e os demais apresentados a seguir, a rede manterá compatibilidade com a arquitetura da Internet e estará de acordo com as tendências regidas pelos grupos RoLL e 6LoWPAN.

Internet estendida: o termo "Internet estendida" refere-se ao posicionamento "central" deste modelo de conectividade em relação à rede de objetos autônomos e a "autêntica" IoT. Em outras palavras, este modelo apresenta as redes de objetos inteligentes parcialmente ou completamente conectadas à Internet, porém com requisitos apropriados de proteção e segurança [Vasseur and Dunkels 2010]. Aplicações para Cidades Inteligentes (*Smart Cities*) são exemplos desse modelo de conectividade. Essas aplicações produzirão informações úteis para seus habitantes terem uma melhor qualidade de vida e tomar decisões importantes diariamente. Para viabilizar as cidades inteligentes, pode-se usar o modelo apresentado na Figura 1.6 (II), em que o acesso à rede de objetos se dá via *firewall* e *proxy*, os quais possuem a tarefa de controle de acesso seguro às redes privadas de objetos inteligentes. Deste modo, o modelo de conectividade "estende" a Internet por permitir o acesso aos objetos inteligentes que até então estavam isolados.

Internet das Coisas: este modelo, no espectro considerado, é o extremo oposto ao mo-

delo de rede autônoma de objetos. Na IoT, os objetos inteligentes estão verdadeiramente conectados à Internet. Deste modo, os objetos podem prover serviços tais como quaisquer outros dispositivos na Internet, por exemplo, um objeto inteligente pode ser um servidor Web. Qualquer usuário da Internet, seja humano ou máquina, poderá ter acesso aos recursos dos objetos inteligentes. Como mostrado na Figura 1.6 (III), o acesso se dá conectando-se diretamente com o objeto ou através de um *proxy*[16]. Estes servidores podem estar localizados dentro ou fora da sub-rede de objetos inteligentes e possuem a tarefa de coletar informações dos dispositivos. Ao usar o *proxy* tem-se que a Internet conecta o usuário ao servidor *proxy*, o qual está conectado aos dispositivos. Assim, técnicas de processamento dentro da rede (*in-networking processing*) podem ser utilizadas, na rede entre o *proxy* e os dispositivos, para preservar os recursos e manter o princípio fim-a-fim.

1.3.5. Paradigmas de comunicação para objetos inteligentes



(a) Muitos- para-Um (b) Um-para-Muitos (c) Mescla de (a) e (b) (d) Um-para-Um

Figura 1.7. Paradigmas de comunicação

A IoT é uma evolução e combinação de diversas tecnologias como discutido na Seção 1.1. Neste sentido, há interseção entre RSSF e IoT no que tange os paradigmas de comunicação. Esta seção aborda tais paradigmas, classificando-os em quatro categorias como mostrado na Figura 1.7. Os paradigmas diferem significativamente, sendo assim o modo de comunicação pode acarretar em maior ou menor impacto no uso dos recursos, em especial de memória, energia e na viabilidade de aplicações sobre a rede. A seguir, cada um dos paradigmas será descrito, bem como exemplificado através de um protocolo que o implementa:

Muitos-para-Um (*Many-to-One*): os objetos inteligentes reportam informações, as quais são coletadas por estações base (raiz na Figura 1.7). Este paradigma é conhecido como coleta de dados, sendo o mais comum dos paradigmas, visto que atende às demandas de comunicação de muitas aplicações. Para realizar a coleta de dados, cria-se uma árvore de roteamento com raiz nas estações base. Em geral, este paradigma não acarreta em grande consumo de memória e energia. Entretanto, aplicações que necessitam de confirmação de entrega de dados são inviabilizadas, pois não existem rotas reversas entre a raiz e os objetos na rede;

[16]Um *proxy* é um servidor (sistema de computador ou uma aplicação) que age como um intermediário para requisições de clientes solicitando recursos de outros servidores

O exemplo que ilustra o estado-da-arte deste paradigma é o *Collection Tree Protocol (CTP)* [Gnawali et al. 2009]. O CTP emprega a métrica *Expected Transmission count (ETX)* [Javaid et al. 2009] e Trickle [Levis et al. 2004] para, respectivamente, criar rotas de alta qualidade e manter tais rotas a baixo custo.

Um-para-Muitos (*One-to-Many*): é conhecido como disseminação de dados, o qual tem a característica reversa do paradigma de coleta de dados. Na disseminação de dados, as estações base comumente enviam comandos para um ou vários objetos da rede. O intuito, em geral, é reconfigurar ou modificar parâmetros dos dispositivos na rede. Para realizar a disseminação de dados pode-se, por exemplo, efetuar inundações na rede para alcançar os dispositivos alvo. Entretanto, não é possível confirmar se os dados enviados foram entregues tal como ocorre com o CTP para a coleta de dados. O custo em número de mensagens também pode ser alto, caso sejam realizadas diversas inundações na rede.

Deluge é um exemplo de protocolo de disseminação [Chlipala et al. 2004]. O protocolo é otimizado para disseminação para grandes quantidades de dados e faz isso utilizando a métrica ETX para encontrar rotas de alta qualidade;

Um-para-Muitos e vice-versa (*One-to-Many and Many-to-One*): é um paradigma que combina os dois anteriormente apresentados. Nesta abordagem, os objetos inteligentes podem se comunicar com as estações base e vice-versa. Isto amplia a gama de aplicações antes não possíveis, tal como protocolos de transporte confiáveis. Entretanto, o paradigma necessita de memória adicional para manter as rotas bi-direcionais.

O *eXtend Collection Tree Protocol (XCTP)* [Santos et al. 2015a] é um exemplo desta categoria. Os autores argumentam que o XCTP é uma extensão do CTP e, por esse motivo, mantém todas as características do CTP ao mesmo tempo que o estende ao viabilizar rotas reversas entre a raiz e os elementos da rede.

Um-para-um (*Any-to-Any*): esse paradigma é o mais geral possível, pois permite que quaisquer dois objetos inteligentes da rede se comuniquem. Por ser mais abrangente, esta abordagem é a mais complexa e, geralmente, exige maior quantidade de recursos, principalmente de armazenamento, pois se faz necessário manter rotas para todos os dispositivos alcançáveis na rede. Por outro lado, não apresenta restrições significativas para as aplicações, exceto se os recursos computacionais dos dispositivos forem bastante limitados.

O principal protocolo de roteamento da Internet das Coisas reside nesta categoria. O protocolo RPL, descrito em detalhes a seguir, é flexível o suficiente para permitir rotas um-para-um, além de possibilitar que elementos de rede com diferentes capacidades sejam empregados para otimizar o armazenamento das rotas. Ainda existe um modo de operação em que, o RPL, opera sob o paradigma muitos-para-um, sendo portanto, um protocolo flexível no que tange as suas opções de operação.

1.3.6. IPv6 Routing Protocol for Low-Power and Lossy Networks

O *Routing Protocol for Low-Power and Lossy Networks* (RPL) é um protocolo de roteamento para LLNs, projetado e padronizado pelo *ROLL Working Group in the IETF*, sendo o protocolo padrão que utiliza IPv6 para LLNs.

1.3.6.1. Grafo Acíclico Direcionado

Dispositivos de rede executando RPL estão conectados de maneira acíclica. Assim, um Grafo Acíclico Direcionado Orientado ao Destino (DODAG, do inglês *Destination-Oriented Directed Acyclic Graph*) é criado, como mostra a Figura 1.8. Cada nó mantém a melhor rota para a raiz do DODAG. Para encontrar a melhor rota os nós usam uma função objetivo (OF) que define a métrica de roteamento (e.g., ETX da rota [Javaid et al. 2009]) a ser computada.

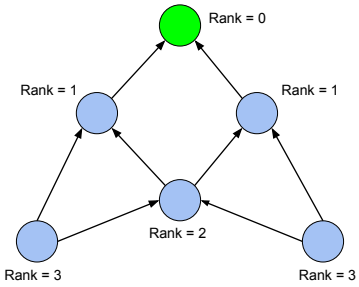


Figura 1.8. DODAG RPL

O RPL utiliza quatro tipos de mensagens de controle, descritas abaixo, para manter e atualizar as rotas. O processo de roteamento inicia pela construção do DAG. A raiz anuncia informações sobre seu DODAG (de um único nó) para todos vizinhos alcançáveis. Em cada nível da árvore de roteamento, os nós tomam decisões sobre rotas baseadas na OF. Uma vez que o nó se junta ao DODAG, ele escolhe uma raiz como seu pai e calcula seu *rank*, que é uma métrica que indica as coordenadas do nó na hierarquia da rede [Vasseur et al. 2011]. Os demais nós irão repetir esse processo de seleção do pai e notificação das informações do DODAG para possíveis novos dispositivos. Quando esse processo se estabiliza, o roteamento de dados pode então começar. O processo descrito cria rotas ascendentes (dos nós para uma raiz). Para construir as rotas descendentes o RPL emite mensagens especiais discutidas a seguir.

1.3.6.2. Mensagens

O RPL especifica três tipos de mensagens utilizando o ICMPv6: *DODAG Destination Advertisement Object* (DAO), *DODAG Information Object* (DIO) e *DODAG Information Solicitation Message* (DIS) descritas a seguir:

DIO: mensagens deste tipo são utilizadas para anunciar um DODAG e suas características. Assim, elas são usadas para a descoberta de DODAGs, sua formação e manutenção. O intervalo de envio de DIO é controlado de modo eficiente pelo algoritmo Trickle [Levis et al. 2004].

DIS: esse tipo de mensagem é similar a mensagens de solicitação de rotas (RS) do IPv6, e são usadas para descobrir DODAGs na vizinhança e solicitar DIOs de nós vizinhos, sem possuir corpo de mensagem.

DAO: mensagens DAO são utilizadas durante o processo de notificação de rotas descendentes. Elas são enviadas em sentido ascendente (dos nós que manifestam o desejo de receber mensagens para seus pais preferenciais) para propagar informações de destino ao longo do DODAG. Essas informações são utilizadas para o preenchimento das tabelas de roteamento descendente, que permitem o tráfego P2MP (ponto a multi-ponto) e P2P (ponto a ponto).

1.3.6.3. Rotas descendentes

As rotas descendentes, da raiz para os nós, são ativadas por meio de mensagens DAO propagadas como *unicast* por meio dos pais em direção à raiz. Essas mensagens contêm informações sobre quais prefixos pertencem a qual roteador RPL e quais prefixos podem ser alcançados através de qual roteador RPL. O protocolo especifica dois modos de operação para o roteamento descendente: *storing* e *non-storing*. Ambos os modos requerem a geração de mensagens DAO, que são enviadas e utilizadas de maneira diferente em cada modo de operação. Os dois modos de operação são descritos a seguir.

Modo *storing*: Neste modo, cada roteador RPL deve armazenar rotas para seus destinos em um DODAG. Essas informações são repassadas dos nós para seus pais preferenciais. Isso faz com que, em nós mais próximos da raiz, o armazenamento necessário seja definido pelo número de destinos na rede. Com isso, a memória necessária em um nó próximo à raiz e um outro distante da raiz pode variar drasticamente, o que faz com que algum tipo de implementação e manutenção administrativa contínua nos dispositivos sejam necessárias, conforme a rede evolui [Clausen et al. 2013]. Entretanto, tal intervenção é inviável, devido ao perfil dinâmico da rede.

Modo *non-storing*: Neste modo, cada nó gera e envia mensagens DAO para a raiz do DODAG. O intervalo no qual o DAO é enviado varia de acordo com a implementação. Entretanto, a especificação do protocolo sugere que esse intervalo seja inversamente proporcional à distância do nó a raiz. Assim, um nó folha gera mais mensagens que um nó intermediário. Após coletar toda informação necessária, a raiz agrega essa informação. Se precisar enviar uma mensagem descendente na rede, a raiz deve utilizar um cabeçalho IPv6 para roteamento de origem (*source routing*). Dessa forma, os nós encaminham a mensagem até que ela alcance seu destino [Tsvetko 2011]. Ou seja, caso os nós não possuam capacidade de armazenamento para operarem no modo *storing*, a rede sofre maior risco de fragmentação e, portanto, perda de pacotes de dados, consumindo a capacidade da rede com o roteamento de origem [Clausen et al. 2013].

1.3.7. Protocolos da camada de aplicações para IoT

O protocolo HTTP é usado na Internet para acessar informações seguindo a estratégia requisição/resposta no paradigma cliente/servidor. O HTTP foi desenvolvido para redes com computadores tipo PC. Diferentemente dos PCs, os dispositivos usados na IoT possuem poder computacional restrito, o que limita a utilização do protocolo HTTP nesses elementos. Para resolver esse problema, foram desenvolvidos dois protocolos da camada de aplicação especificamente para recuperar informações de dispositivos com baixo poder computacional: CoAP e MQTT.

O *Constrained Application Protocol (CoAP)* é definido e mantido pelo *IETF Constrained RESTful Environments (CoRE) working group*[17]. O CoAP define uma forma de transferir dados tal como é feito através do *REpresentational State Transfer (REST)* e, para tanto, utiliza funcionalidades similares ao do HTTP tais como: *GET*, *POST*, *PUT*, *DELETE*. REST permite que clientes e servidores acessem ou consumam serviços Web de maneira fácil usando *Uniform Resource Identifiers (URIs)*. O CoAP é diferente do

[17] https://datatracker.ietf.org/doc/charter-ietf-core/

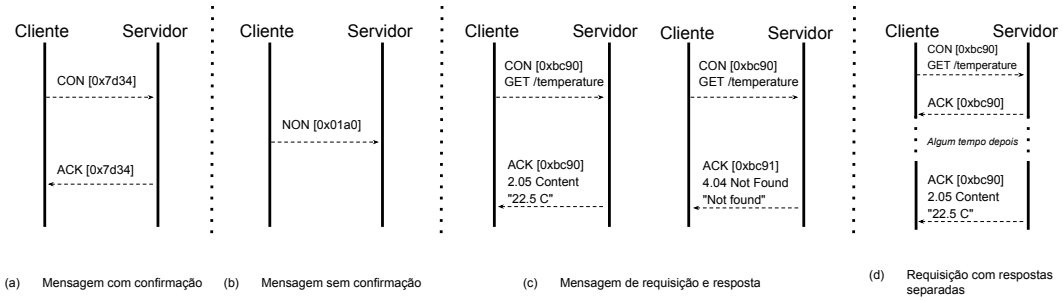


Figura 1.9. Tipos de mensagem do CoAP

REST por usar o protocolo UDP, o que o coloca como mais adequado para aplicações em IoT.

O CoAP apresenta duas camadas em sua arquitetura interna, permitindo que dispositivos de baixa potência possam interagir através de RESTfull. A primeira camada implementa os mecanismos de requisição/resposta. A segunda, detecta mensagens duplicadas e fornece comunicação confiável sobre o UDP. O CoAP utiliza quatro tipos de mensagens (Figura 1.9): *confirmable*, *non-confirmable*, *reset* e *acknowledgement*. A confiabilidade do protocolo CoAP é conseguida ao combinar as mensagens *confirmable* e *non-confirmable* sobre o UDP.

As URIs CoAP são utilizadas da seguinte forma `coap://URI`. Para facilitar o acesso aos recursos, existem algumas extensões que são utilizadas para integrar o CoAP aos *browsers*, por exemplo, a Copper para Firefox. Outras informações podem ser encontradas na RFC 7252[18] e o conjunto de implementações existentes podem ser encontradas no site de um dos autores do CoAP[19].

O *Message Queue Telemetry Transport*(MQTT) é um protocolo projetado para dispositivos extremamente limitados e utiliza a estratégia de *publish/subscribe* para transferir mensagens. O principal objetivo do MQTT é minimizar o uso de largura de banda da rede e recursos dos dispositivos. Além disso, o MQTT provê mecanismos para a garantia de entrega de mensagens. MQTT utiliza os protocolos das camada de transporte e rede da arquitetura TCP/IP. O cabeçalho do protocolo MQTT pode ter tamanho fixo (dois bytes) ou variável. Um exemplo de uma implementação *open source* do MQTT é o Mosquitto[20].

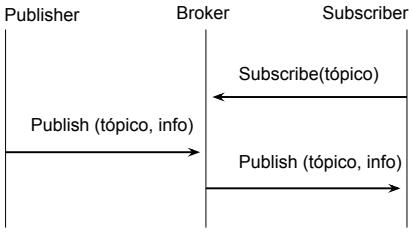


Figura 1.10. *publish/subscribe*

O MQTT consiste de três componentes básicos: o *subscriber*, *o publisher* e o *broker*. A Figura 1.10 mostra a ordem de operação do MQTT. Inicialmente dispositivos

[18] https://tools.ietf.org/html/rfc7252
[19] http://coap.technology
[20] http://mosquitto.org

se registram (*subscribe*) a um *broker* para obter informações sobre dados específicos, para que o *broker* os avise sempre que publicadores (*publishers*) publicarem os dados de interesse. Os dispositivos inteligentes (*publishers*) transmitem informações para os *subscriber* através do *broker*.

1.3.8. Ambientes de desenvolvimento

Assim como softwares para computadores de propósito geral, o software que executa no microcontrolador dentro de um objeto inteligente também é escrito em uma linguagem de programação e compilado para o código de máquina para o microcontrolador. O código de máquina é escrito na ROM do microcontrolador e quando o objeto inteligente é ligado, esse código é executado [Vasseur and Dunkels 2010].

1.3.8.1. Sistemas Operacionais

Assim como os PCs, os objetos inteligentes também utilizam sistemas operacionais. Entretanto, como os objetos inteligentes possuem recursos físicos limitados, os SOs para esses objetos devem ser muito menores e consumir menos recursos. Esta seção descreve brevemente os dois principais sistemas operacionais para objetos inteligentes: Contiki e TinyOS, além do sistema operacional Android que opera em grande parte dos *smartphones* e algumas variações do sistema Linux orientadas à IoT. Todos esses sistemas operacionais são de código aberto. Ao final desta seção, a Tabela 1.2 apresenta uma comparação entre os principais sistemas apresentados.

Contiki [21]**:** é um SO leve para sistemas embarcados de rede, que fornece mecanismos para o desenvolvimento de softwares para IoT e mecanismos de comunicação que permitem a comunicação dos dispositivos inteligentes. Além disso, o Contiki também fornece bibliotecas para a alocação de memória, abstrações de comunicação e mecanismos de redes de rádios de baixa potência. O Contiki foi o primeiro SO para objetos inteligentes a fornecer comunicação IP com a pilha μIP TCP/IP e, em 2008, o sistema incorporou o μIPv6, a menor pilha IPv6. Por utilizar IP, o Contiki pode se comunicar diretamente com outros aplicativos baseados em IP e serviços Wweb [Vasseur and Dunkels 2010]. Tanto o sistema operacional quanto suas aplicações são implementados na linguagem C, o que faz o sistema ser altamente portável [Dunkels et al. 2004].

TinyOS[22]**:** Assim como o Contiki, o TinyOS é um SO para redes de sensores e objetos inteligentes. Um programa do TinyOS é descrito em [Levis et al. 2005] como um grafo de componentes, cada um sendo uma entidade computacional independente que expõe uma ou mais interfaces. Os componentes podem ser chamados comandos (requisições a um componente para executar algum serviço), eventos (sinalizam a finalização desse serviço) ou tarefas (usadas para expressar a concorrência intra-componente). TinyOS possui um modelo de programação baseado em componentes, codificado pela linguagem NesC, um dialeto da linguagem C. O modelo de programação fornecido em NesC se baseia em componentes que encapsulam um conjunto específico de serviços, e se comunicam por meio de interfaces.

[21] https://github.com/contiki-os
[22] https://github.com/tinyos

Sistema	Min. RAM	Min. ROM	Linguagem
Contiki	< 2 KB	< 30 KB	C
TinyOS	< 1 KB	< 4 KB	nesC e oTcl
RIOT	∼ 1.5 KB	∼ 5 KB	C e C++
Snappy	128 MB	–	Python, C/C++, Node JS e outras
Raspbian	256 MB	–	Python, C/C++, Node JS e outras

Tabela 1.2. Comparativo entre os principais SOs para IoT

Android[23]**:** é uma plataforma para dispositivos móveis que inclui um SO, *middleware* e aplicativos[24]. Os vários componentes do SO são projetados como uma pilha, com o núcleo do Linux na camada mais baixa. Acima da camada do Linux, estão as bibliotecas nativas do Android, o que permite ao dispositivo manusear diferentes tipos de dados. Na mesma camada das bibliotecas está o *Android Runtime*, que possui um tipo de máquina virtual Java utilizada para a execução de aplicativos no dispositivo Android. A próxima camada está relacionada com o *framework* de aplicação, que fornece vários serviços de alto nível ou APIs para os desenvolvedores de aplicativos. Por fim, no topo da pilha, está a camada de aplicação.

Linux: com a popularização da IoT, alguns SOs baseados em Linux foram desenvolvidos. o RIOT [25] [26] foi desenvolvido por uma comunidade formada por empresas, acadêmicos e amadores, distribuídos em todo o mundo. Essa plataforma executa em 8, 16 ou 32 bits e possui suporte às linguagens C e C++. Além disso, possui suporte para IoT com implementações 6LoWPAN, IPv6, RPL, e UDP. O Ubuntu também possui sua versão IoT, chamada Ubuntu Core ou Snappy [27]. Essa versão é reduzida em comparação ao Ubuntu *desktop*, uma vez que exige apenas 128 MB de memória RAM e um processador de 600 MHz. O desenvolvimento pode ser feito em diversas linguagens, como o de uma aplicação Linux comum. Raspian [28] é um SO baseado no Debian e otimizado para o hardware do Raspberry Pi. Oferece mais de 35 mil pacotes *deb* de software, que estão pré-compilados, para serem facilmente instalados no Raspberry Pi. O sistema está disponível em três versões: Raspbian Wheezy, DRaspbian Jessie e Raspbian Jessie Lite.

1.3.8.2. Emuladores e simuladores

Simulações e emulações são muito úteis durante a fase de avaliação de arquiteturas e protocolos para redes em geral. Esses ambientes de desenvolvimento permitem modelar uma rede de computadores arbitrária, especificando o comportamento dos elementos da rede, bem como os enlaces de comunicação. Esta seção apresenta os principais

[23]O Android é disponibilizado sob licença de código aberto, apesar da maior parte dos dispositivos apresentarem uma combinação de software livre e privado: *https://github.com/android*

[24]http://developer.android.com/guide/basics/what-is-android

[25]http://www.riot-os.org/

[26]https://github.com/RIOT-OS/RIOT

[27]http://www.ubuntu.com/internet-of-things

[28]https://www.raspbian.org/

simuladores e emuladores de redes de computadores que possuem suporte para IoT.

Essencialmente, há diferenças entre os emuladores e simuladores como é destacado em [Bagula and Erasmus 2015]. Emulador é um sistema de hardware e/ou software que permite a um sistema de computador (chamado de *host*) se comportar como um outro sistema de computador (chamado de *guest*). Em outras palavras, o emulador se comporta exatamente como o *guest* permitindo ao *host* executar um software projetado para o sistema *guest*. Por exemplo, o emulador Cooja do Contiki permite a um computador se comportar como um dispositivo inteligente Tmote Sky[29]. Já o simulador, por sua vez, é modela ("imita") uma situação da real ou hipotética em um computador. Com a simulação é possível estudar como o sistema simulado deve operar, caso o modelo reflita as características do mundo real.

A seguir, são apresentadas algumas dessas plataformas.

Cooja: é um emulador de aplicações do sistema operacional Contiki, desenvolvido na linguagem Java (Cooja – **Co**ntiki **OS** **Ja**va). As simulações no Cooja consistem em cenários onde cada nó possui um tipo, memória e um número de interfaces [Österlind]. Um nó no Cooja é um sistema Contiki real compilado e executado. Esse sistema é controlado e analisado pelo Cooja [30], que possui uma interface para analisar e interagir com os nós, o que facilita o trabalho e a visualização da rede. Além disso, é possível criar cenários personalizados.

ns-2/ns-3: Ns-2 é um simulador de eventos discretos para rede de computadores de código aberto. As simulações para ns-2 são compostas por código em C++ e *scrips* oTcl. O código é utilizado para modelar o comportamento dos nós, enquanto os *scripts* controlam a simulação e especificam aspectos adicionais, como a topologia da rede. O ns-3 também é um simulador de eventos discretos, mas não é uma extensão de seu antecessor, ns-2. Assim como o ns-2, o ns-3 adota a linguagem C++ para a implementação dos códigos, mas não utiliza mais scripts oTcl. Com isso, as simulações podem ser implementadas em C++, com partes opcionais implementadas usando Python [Weingärtner et al. 2009].

Tossim: é o simulador de eventos discretos do SO TinyOS. Ao invés de compilar uma aplicação TinyOS para um nó sensor, os usuários podem compilá-lo para o TOSSIM, que é executado em um PC. No Tossim é possível simular milhares de nós, cada nó, na simulação, executa o mesmo programa TinyOS. O simulador se concentra em simular o TinyOS e sua execução, ao invés de simular o mundo real. Por isso, apesar de poder ser usado para entender as causas do comportamento observado no mundo real, não é capaz de capturar todos eles e pode não ser o mais fidedigno para avaliações [Levis and Lee 2010]. Na abstração do Tossim, a rede é um grafo orientado no qual cada vértice é um nó e cada aresta (enlace) tem uma probabilidade de erro de bit. O simulador fornece um conjunto de serviços que permitem interação com aplicativos externos [Levis et al. 2003].

OMNet++/Castalia: é um simulador de eventos discretos para modelar redes de comunicação, multiprocessadores e outros sistemas distribuídos ou paralelos [Varga and Hornig 2008]. O OMNet++ fornece o ferramental para criar simulações

[29] http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf

[30] https://github.com/contiki-os/contiki/wiki/An-Introduction-to-Cooja

Nome	Suporte GUI	Licença	Linguagem	Sistema Operacional
Cooja	Sim	Código aberto	C	Linux
ns-2	Não	Código aberto	C++ e oTcl	GNU/Linux, FreeBSD, Mac OS e Windows
ns-3	Limitado	Código aberto	C++ e python	GNU/Linux, FreeBSD, Mac OS e Windows
Tossim	Limitado	Código aberto	nesC e python	Linux ou Cygwin no Windows
OMNet++	Sim	Comercial e código aberto	C++	Linux, Mac OS e Windows
Castalia	Sim	Código aberto	C++	Linux e Windows
Sinalgo	Sim	Código aberto	Java	Linux, Mac OS e Windows

Tabela 1.3. Comparativo entre os principais simuladores/emuladores para IoT

de diferentes tipos de redes, tendo uma IDE baseada no Eclipse, um ambiente de execução gráfica e outras funcionalidades. Existem extensões para simulação em tempo real, emulação de rede, integração de banco de dados, integração de sistemas, entre outras funções [Varga et al. 2001]. Castalia é um simulador para RSSFs e outras redes de dispositivos embarcados de baixa potência. Ele é baseado na plataforma do OMNeT++ e pode ser usado para pesquisas e desenvolvimento que requerem testar algoritmos distribuídos e/ou protocolos em modelos realistas de canais sem fio e rádio [Boulis et al. 2011].

Sinalgo: é um *framework* para avaliar algoritmos em rede (Sinalgo (**Si**mulator for **N**etwork **Algo**rithms)), abstraindo a pilha de protocolos de comunicação entre os elementos computacionais. Assim, o Sinalgo foca no comportamento de algoritmos em rede. Apesar de ter sido desenvolvido para simulação de redes sem fio, pode ser usado para redes com fio. O Sinalgo utiliza a linguagem JAVA, o que torna o desenvolvimento mais fácil e rápido, além de facilitar o processo de depuração[31].

1.3.8.3. Testbeds

Testbed é uma plataforma para implantar aplicações em um contexto real, ou seja, utilizando hardware real em larga escala, apropriada para a gestão de experimentação. Atualmente, existem vários *testbeds* que permitem a experimentação mais rápida. A lista, a seguir, apresenta os principais *testbeds* para a experimentação em IoT.

WHY-NET: é um *testbed* escalável para redes sem fio móveis. Possui diferentes tecnologias sem fio como Redes de Sensores e Antenas inteligentes [Patel and Rutvij H. 2015].

ORBIT: consiste de 400 nós de rádio fixos. Cada nó físico está logicamente ligado a um nó virtual em uma rede. Os nós de rádio possuem duas interfaces: IEEE 802.11x e Bluetooth. As medições podem ser realizados no nível físico, MAC e rede [Patel and Rutvij H. 2015].

[31] http://dcg.ethz.ch/projects/sinalgo/

MiNT: neste *testbed*, o nó de rádio IEEE 802.11 é conectado a cada robô de controle remoto. Para evitar fontes de ruído, o *testbed* é configurado em uma única sala de laboratório. MiNT suporta reconfiguração de topologia e mobilidade irrestrita do nó [Patel and Rutvij H. 2015].

IoT-LAB: oferece uma infraestrutura de grande escala adequada para testar pequenos dispositivos sensores sem fios e comunicação de objetos heterogêneos. Também oferece ferramentas Web para desenvolvimento de aplicações, juntamente com o acesso de linha de comando direto à plataforma. *Firmwares* de nós sensores podem ser construídas a partir da fonte e implantado em nós reservados. A atividade de aplicação pode ser controlada e monitorada, o consumo de energia e interferência de rádio podem ser medidos usando as ferramentas fornecidas[32].

Indriya: é um *testbed* de rede de sensores sem fio, que possui 127 motes TelosB. Mais de 50% dos motes são equipados com diferentes módulos de sensores, incluindo infravermelho passivo (PIR), magnetômetro e acelerômetro. Indriya usa o software de interface do usuário do Motelab, que fornece acesso baseado na Web para os nós do *testbed*, que está localizado na Universidade Nacional de Singapura [Doddavenkatappa et al. 2012].

1.3.9. Gateway

Na IoT, é comum que os dispositivos apresentem tecnologias de comunicação heterogêneas, por exemplo BLE, ZigBee, e outros. Para conectar esses dispositivos à Internet, é preciso que um elemento de rede realize a tradução entre os diversas tecnologias utilizadas, este elemento é chamado de *gateway*. Na Figura 1.11 são apresentadas duas visões possíveis do *gateway*. Na parte superior da figura, o *gateway* realiza a tradução de tecnologias distintas e o encaminhamento de mensagens. Neste caso, o *gateway* respeita o princípio fim-a-fim, porém a complexidade de hardware/software e gerenciamento aumentam. Por exemplo, o *gateway* precisa implementar todas as interfaces de comunicação da sub-rede IoT, bem como a interface que o conecta à Internet. Além disso, é preciso um software para controlar e gerenciar as regras de operação da rede.

O *gateway* também pode ser visto como um prestador de serviços da rede. Um exemplo pode ser identificado na parte inferior da Figura 1.11. Nesse caso, o *gateway* funciona como um *proxy*, realizando compressão transparente para que aplicações que utilizem protocolo IP não necessitem de modificações. Um local típico para alocação de um *proxy* seria no *gateway*, ou em um servidor proxy local [Shelby and Bormann 2011].

1.3.10. Segurança

Para que um sistema IoT seja seguro é preciso estabelecer quais são os objetivos de segurança desejáveis. Existem pelo menos três grupos de objetivos desejáveis para segurança em IoT [Shelby and Bormann 2011]: (i) *confidencialidade* – requisito onde os dados transmitidos podem ser "escutados" e entendidos por elementos participantes da comunicação, isto é, elementos sem autorização sabem que ocorreu comunicação, mas não sabem o conteúdo da comunicação; (ii) *integridade* – os dados não podem ser alterados por elementos da rede sem devida autorização. É comum que *hackers* adulterem mensagens sem deixar vestígios e a quebra da integridade passe despercebida. De modo

[32] https://www.iot-lab.info

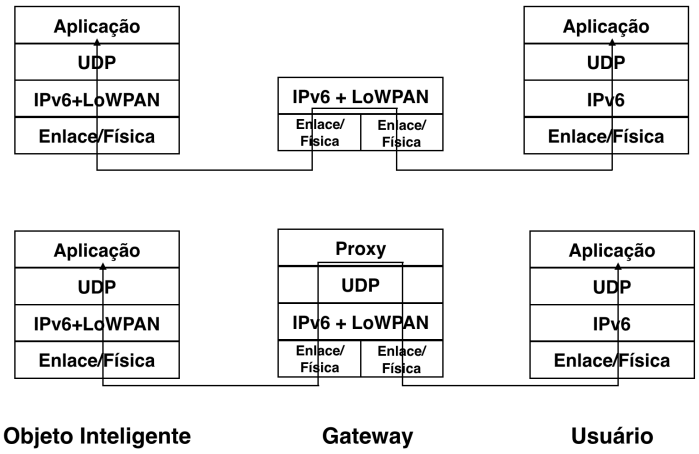


Figura 1.11. Pilha de protocolos de um *gateway*: Comunicação fim-a-fim (superior); Comunicação através de um *proxy* (inferior)

geral, implementa-se integridade criptografando as mensagens e verificando-as no lado do receptor; (iii) *disponibilidade* – deseja-se manter o sistema sempre disponível e seguros contra ataques maliciosos. Entretanto, as redes sem fio estão sujeitas a interferências de comunicação e "hackers" podem agir nesta vulnerabilidade. Assim, o sistema IoT deve ser capaz de identificar e tratar problemas como este para evitar ataques *Denial of Service (DoS)*.

O modelo de ameaças (*threat model*) da IoT (6LoWPAN) é semelhante ao utilizado nos protocolos da Internet[33]. Assume-se que o "hacker" possui controle sobre a rede podendo ler, alterar ou remover qualquer mensagem na rede. Portanto, sem o suporte à criptografia torna-se inviável proteger ou detectar adulterações indevidas no sistema. O suporte à segurança pode ser implementado em diferentes camadas da pilha de protocolos. Por exemplo, na camada de enlace pode-se aplicar o algoritmo *Advanced Encryption Standard (AES)* em conjunto com *Counter with CBC-MAC (CCM)*[34] para manter confidencialidade a cada salto na rota entre os comunicantes. Entretanto, esta técnica não oferece qualquer segurança sobre a integridade dos dados. Por outro lado, na camada de rede pode-se empregar o IPsec[35] para alcançar integridade. No entanto, o padrão IPsec é considerado "pesado" para as restrições dos dispositivos IoT e, assim, é preciso adaptá-lo. Vale mencionar que os requisitos de segurança para IoT variam de aplicação para aplicação e, assim, devem considerar um ou mais dos objetivos de segurança acima mencionados ao implementar uma aplicação.

1.3.11. Desafios e questões de pesquisa

A Internet das Coisas apresenta vários desafios de implementação, como os discutidos acima. Esses desafios já levaram e continuam a levar à investigação de várias

[33]O RFC 3552 apresenta boas práticas e discussões acerca de segurança em IoT: https://tools.ietf.org/html/rfc3552

[34]RFC 3610 https://www.ietf.org/rfc/rfc3610.txt

[35]RFC 4301: https://tools.ietf.org/html/rfc4301

questões de pesquisa decorrentes das restrições dos dispositivos existentes na IoT, dos protocolos e soluções de interconexão desses dispositivos e da escalabilidade da rede. Além disso, à medida que a Internet das Coisas se tornar cada vez mais presente na sociedade, o projeto de aplicações e serviços, que estão sendo propostos para os diversos segmentos das atividades humana, trará também novas questões de pesquisa relacionadas principalmente ao tratamento de dados coletados por esses dispositivos. Em outras palavras, essa é uma área extremamente fértil para explorar novas questões de pesquisa que têm o potencial de impactar o estado-da-arte.

1.4. IoT na prática

As seções anteriores trouxeram uma visão geral sobre as bases da IoT. Foram discutidas diversas características dos objetos inteligentes permeando particularidades teóricas e artefatos existentes já empregados na IoT. Já esta seção traz uma perspectiva prática da IoT. Diante do que já foi discutido, o leitor está apto a por em prática os conceitos vistos. Os exercícios práticos visam consolidar e associar os conceitos teóricos e artefatos previamente discutidos. Além disso, uma das práticas servirá de âncora para assuntos das próximas seções, vinculando as redes de objetos inteligentes, até aqui apresentadas, com os desafios e próximos passos em relação ao futuro da IoT.

Os exemplos apresentados, a seguir, foram extraídos do conjunto de exemplos do sistema operacional Contiki versão 3.0. Presume-se que o leitor já tenha o Contiki instalado e operacional[36]. Todos os exemplos são de código livre e hospedados no repositório oficial do Contiki. Inicialmente será exemplificado como conectar os objetos inteligentes utilizando IPv6[37] e, em seguida, será pleiteado o Erbium que é uma implementação do CoAP[38] para redes de objetos de baixa potência no Contiki.

Os experimentos apresentados visam atender o maior publico possível. Para isto, ao longo do texto a prática é exemplificada através do uso de simulador. Entretanto, o minicurso também apresenta uma página Web[39] onde podem ser encontrados materiais extra preparados pelos autores. Nesse endereço, encontram-se vídeo tutoriais, textos, referências e outros conteúdos relacionados.

1.4.1. Rede de objetos inteligentes IPv6

No primeiro experimento prático, será criada uma rede IPv6 de objetos inteligentes utilizando Cooja (veja a Seção 1.3.8). O Contiki oferece uma implementação do 6LoWPAN em conjunto com o protocolo de roteamento RPL [Hui 2012]. Além disso, o SO também oferece suporte à pilha de protocolos μIP TCP/IP (veja Seção 1.3.3). Para começar, inicie o Cooja através do atalho da área de trabalho ou execute o comando abaixo e, em seguida, crie uma nova simulação:

[36] http://www.contiki-os.org/start.html

[37] https://github.com/contiki-os/contiki/tree/master/examples/ipv6/rpl-border-router

[38] https://github.com/contiki-os/contiki/tree/master/examples/er-rest-example

[39] **http://homepages.dcc.ufmg.br/~bruno.ps/iot-tp-sbrc-2016/**

Inicie o Cooja executando os comandos abaixo:

```
$ cd <DIR>+contiki/tools/cooja/
$ ant run
```

Dando seguimento à prática, dispositivos *Tmote Sky* serão emulados. Um dos *motes* servirá como roteador de borda da rede de objetos IPv6. O roteador de borda é o dispositivo responsável por configurar um prefixo de rede e iniciar a construção da árvore de roteamento do RPL. Em outras palavras, o roteador de borda nada mais é que a raiz da árvore de roteamento e interlocutor entre a rede de objetos e a rede externa.

O código do roteador de borda está localizado em:

```
<DIR>+contiki/examples/ipv6/rpl-border-router/border-router.c
```

Com o Cooja aberto, vá até a aba de criação de *mote* e adicione um *Sky mote*. Na tela seguinte, compile e carregue, como *firmeware*, o código do roteador de borda. Finalmente adicione **1 *mote*** deste tipo.

Após configurar o roteador de borda, a próxima etapa é povoar a rede com objetos inteligentes. O código *sky-websense.c* ajudará nesta fase.

O código do *sky-websense.c* está localizado em:

```
<DIR>+contiki/examples/ipv6/sky-websense/sky-websense.c
```

Este código é uma aplicação que produz "dados sensoriados" e prover acesso a estas informações via *webserver*. Adicione alguns[40] *motes* desse tipo. É recomendável que o o leitor investigue o código *sky-websense.c* para entender o que ele faz. Retorne para o Cooja. Na aba chamada *"Network"*, localize o roteador de borda e no seu menu de contexto selecione a opção mostrada abaixo. O Cooja informará que o roteador de borda criou um *socket* e está escutando na porta local `60001`. Em seguida inicialize a simulação.

No Menu-Contexto do Roteador de Borda selecione:

"Mote tools for sky/Serial socket (SERVER)"

Abra um novo terminal e execute os seguintes comandos:

Comandos para executar o programa `tunslip6`

```
$ cd <DIR>+contiki/examples/ipv6/rpl-border-router/
$ make connect-router-cooja TARGET=sky
```

Estes comandos acabam por executar o programa `tunslip6`. O programa configura uma interface na pilha de protocolos do Linux, em seguida, conecta a interface ao *socket* em que o roteador de borda está escutando. O `tunslip6` fará com que todo o tráfego endereçado ao prefixo **aaaa:**[41] seja direcionado para a interface do Cooja.

[40] Adicione 2 ou 5 *motes* para manter a carga de processamento e consumo de memória baixos.

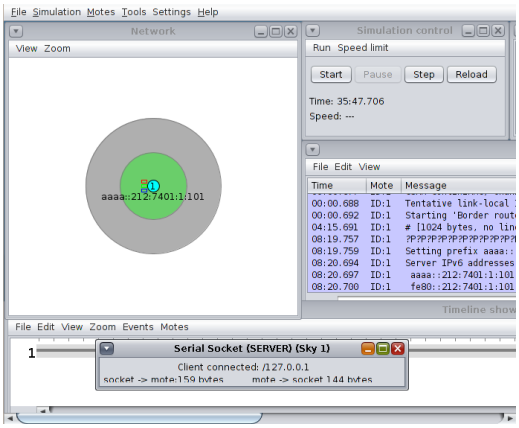
[41] Vale ressaltar que os endereços aqui apresentados serão expressos em modo comprimido. Por exemplo, `aaaa:0000:0000:0000:0212:7401:0001:0101` é o mesmo que `aaaa::212:7401:1:101`

Terminal 2 – `tunslip6`

```
ifconfig tun0 inet `hostname` up
ifconfig tun0 add aaaa::1/64
ifconfig tun0 add fe80::0:0:0:1/64
ifconfig tun0

tun0 Link encap:UNSPEC  HWaddr 00-00...
     inet addr:127.0.1.1  P-t-P:127.0.1.1
     Mask:255.255.255.255
     inet6 addr: fe80::1/64 Scope:Link
     inet6 addr: aaaa::1/64 Scope:Global
     ...

Address:aaaa::1 => aaaa:0000:0000:0000
Got configuration message of type P
Setting prefix aaaa::
Server IPv6 addresses:
 aaaa::212:7401:1:101
 fe80::212:7401:1:101
```



O `tunslip6` apresentará uma saída semelhante a mostrada na caixa de título Terminal 2 e o Cooja apresentará algo como mostrado na figura ao lado[42]. Agora já é possível verificar se os *Tmotes Sky* emulados estão acessíveis através de ferramentas como o `ping6`.

Realize testes com os seguintes comandos:

`$ping6 aaaa::212:7401:1:101` (Roteador de borda)
`$ping6 aaaa::212:7402:2:202` (*Tmote* rodando *sky-websense*)

Em seu navegador, acesse o endereço do roteador de borda http://[aaaa::212:7401:1:101]/

A rede de exemplo possui três *motes* e apresenta topologia exibida na figura abaixo, em que ~:101 é o roteador de borda.

O Roteador de Borda responderá com:

```
Neighbors

fe80::212:7402:2:202

Routes

aaaa::212:7402:2:202/128 via fe80::212:7402:2:202 16711412s
aaaa::212:7403:3:303/128 via fe80::212:7402:2:202 16711418s
```

A resposta à requisição feita ao roteador de borda conterá duas partes. A primeira exibe a tabela de vizinhança, ou seja, os nós diretamente ligados na árvore de roteamento do RPL (*Neighbor Table*). A segunda lista os nós alcançáveis (*Routes*) e o próximo salto na rota até aquele destinatário.

Agora acesse, pelo navegador, os recursos (luminosidade e temperatura) providos pelos *Tmotes* rodado *sky-websense* como mostrado a seguir:

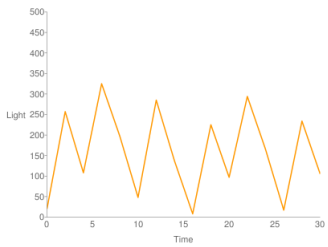
Acesse os *Tmotes* pelo *browser* com:

`http://[IPv6 do Tmote]/`
`http://[IPv6 do Tmote]/l`
`http://[IPv6 do Tmote]/t`

Exemplo de requisição:

`http://[aaaa::212:7403:3:303]/l`

Exemplo de resposta:
`http://[aaaa::212:7403:3:303]/l`



[42]Note que os *motes* executando *websense* não foram exibidos.

1.4.2. Erbium (Er) REST: Uma implementação CoAP no Contiki-OS

Esta prática abordará o uso de *Representational State Transfer* (REST) – Transferência de Estado Representacional. Para tanto, será utilizado o Erbium (Er), uma implementação do IETF CoAP de RTF 7252 (veja a Seção 1.3.7). O Er foi projetado para operar em dispositivos de baixa potência [Kovatsch et al. 2011] e tem código livre disponível junto com o sistema operacional Contiki. Para iniciar será feita uma breve revisão da implementação e uso do CoAP no Contiki.

1.4.2.1. CoAP API no Contiki

No CoAP, os serviços disponibilizados pelo servidor são vistos como recursos, os quais são identificados por URIs únicas. O CoAP oferece diferentes métodos para realizar operações básicas CRUD sendo eles: **POST** para criar um recurso; **GET** para recuperar um recurso; **PUT** para atualizar algum recurso e; **DELETE** para apagar um recurso.

A implementação do CoAP no Contiki está localizada no diretório:

```
<DIR>+contiki/apps/er-coap<version>
```

Para cada recurso disponibilizado no servidor usando CoAP existe uma função (*handle function*), a qual a aplicação REST chama sempre que ocorre uma requisição de um cliente. Com a *handlefunction*, o servidor é capaz de tratar cada requisição e responder adequadamente com o conteúdo do recurso solicitado.

As macros[43] a seguir são providas pela implementação do CoAP/Erbium do Contiki para facilitar a criação de novos recursos para o servidor:

Normal Resource: este tipo de recurso serve de base para todos as demais macros. O *Normal Resource* associa uma URI a uma *handle function*.

```
1 #define RESOURCE(name, attributes, get_handler, post_handler, put_handler,
      delete_handler)
```

Parent Resource: destinado a gerenciar diversos sub-recursos de uma URI. Por exemplo, a URI test/parent/<sub-recursos>.

```
1 #define PARENT_RESOURCE(name, attributes, get_handler, post_handler, put_handler,
      delete_handler)
```

Separate Resource: esta macro é bastante utilizada quando é necessário enviar informações em partes. Por exemplo, quando se tem algum arquivo na memória do dispositivo. Deste modo, o arquivo pode ser enviado em partes separadas para o cliente que o solicitou.

```
1 #define SEPARATE_RESOURCE(name, attributes, get_handler, post_handler, put_handler,
      delete_handler, resume_handler)
```

Event Resource e **Periodic Resource**: no primeiro, a ideia é que quando um evento ocorra o dispositivo envie uma mensagem para algum cliente que esteja "observando" aquele recurso, por exemplo, quando um botão no dispositivo é pressionado envia-se uma

[43]Mais detalhes sobre as macros, atributos e estruturas são encontrados em: https://github.com/contiki-os/contiki/tree/master/apps/er-coap

mensagem para o cliente. No segundo, existe uma periodicidade no envio das mensagens, para isto um parâmetro extra indica o período. Vale pontuar que os dois recursos são *observáveis*, isto é, um cliente ao "assinar" o *feed* daquele recurso receberá notificações sempre que ocorrer mudanças naquele recurso.

```
#define EVENT_RESOURCE(name, attributes, get_handler, post_handler, put_handler,
    delete_handler, event_handler)
```

```
#define PERIODIC_RESOURCE(name, attributes, get_handler, post_handler, put_handler,
    delete_handler, period, periodic_handler)
```

Para exemplificar a implementação de um recurso, será tomado como modelo o recurso *helloworld* do arquivo *er-example-server.c* do conjunto de exemplo do Er. Abaixo é exibido um fragmento do arquivo e alguns comentários traduzidos.

```
1  /*
2   * Assinatura do metodo: nome do recurso, metodo que o recurso manipula e URI.
3   */
4  RESOURCE(helloworld, METHOD_GET, "hello", "title=\"Hello world ?len=0..\";rt=\"Text\"");
5
6  /*
7   * Handleer function [nome do recurso]_handle (Deve ser implementado para cada recurso)
8   * Um ponteiro para buffer, no qual a conteudo (payload) da resposta sera anexado.
9   * Recursos simples podem ignorar os parametros preferred_size e offset, mas devem
10  * respeitar REST_MAX_CHUNK_SIZE (limite do buffer).
11  */
12 void helloworld_handler(void* request, void* response, uint8_t *buffer, uint16_t
        preferred_size, int32_t *offset) {
13   const char *len = NULL;
14   char const * const message = "Hello World!";
15   int length = 12; /*           |<---------->| */
16
17   /* A URI solicitada pode ser recuperada usando rest_get_query() ou usando um parser
        que retorna chave-valor. */
18   if (REST.get_query_variable(request, "len", &len)) {
19     length = atoi(len);
20     if (length<0) length = 0;
21   if (length>REST_MAX_CHUNK_SIZE) length = REST_MAX_CHUNK_SIZE;
22     memcpy(buffer, message, length);
23   } else {
24     memcpy(buffer, message, length);
25   }
26
27   REST.set_header_content_type(response, REST.type.TEXT_PLAIN);
28   REST.set_header_etag(response, (uint8_t *) &length, 1);
29   REST.set_response_payload(response, buffer, length);
30 }
31
32 /* Inicializa a REST engine. */
33 rest_init_engine();
34
35 /* Habilita o recurso. */
36 rest_activate_resource(&helloworld);
```

- Na linha 4 é declarado um *Normal Resource*, indicando o nome do recurso, bem como o método que o recurso aceita (pode aceitar mais de um método), seguidos da URI e de um texto de descrição.
- Na linha 12 é declarada a função que manipula as requisições para a URI do recurso. O padrão `[nome do recurso]_handler` deve ser mantido. Ao ser invocada, a função envia uma mensagem *"Hello World"* para o cliente. Além disso, se o parâmetro *"len"* for passado e o valor for menor que o comprimento da *string*

"Hello World", então a função retorna somente os caracteres `[0:len]`. Se o parâmetro contiver valor 0, então a função retorna uma *string* vazia.

- Entre as linhas 18 e 25 o processamento da requisição é realizado e o *buffer* de resposta é preenchido adequadamente.

- Nas linhas 27 e 29 o cabeçalho da mensagem de resposta é preenchido indicando respectivamente o tipo da mensagem (`TEXT_PLAIN`) e o comprimento da mensagem. Finalmente na linha 31 a carga da mensagem é preenchida.

- Uma vez que o recurso já foi declarado e implementado é preciso inicializar o *framework* REST e iniciar o processo CoAP, isto é realizado na linha 33. Além disso, é preciso habilitar o recurso para que ele esteja disponível para o acesso via URI, isto é feito na linha 36.

Este é um esboço da implementação de um recurso. É recomendável a leitura dos arquivos citados, bem como o material extra do minicurso para completo entendimento.

1.4.2.2. CoAP Erbium Contiki

Nesta etapa será apresentado o uso do Erbium Contiki.

Mude para o seguinte diretório:

```
<DIR>+/contiki/examples/er-rest-example/
```

Neste diretório existe uma conjunto de arquivos de exemplo do uso do Erbium[44]. Por exemplo, o arquivo *er-example-server.c* mostra como desenvolver aplicações REST do lado servidor. Já *er-example-client.c* mostra como desenvolver um cliente que consulta um determinado recurso do servidor a cada 10s. Antes de iniciar a prática é conveniente realizar algumas configurações. A primeira delas diz respeito aos endereços que serão utilizados. Deste modo, realize as operações abaixo:

Defina os endereços dos *Tmotes* no arquivo `/etc/hosts`

Adicione os seguintes mapeamentos:

```
aaaa::0212:7401:0001:0101 cooja1
aaaa::0212:7402:0002:0202 cooja2
...
```

Além disso, adicione a extensão Copper (CU)[45] CoAP *user-agent* no Mozilla Firefox.

No arquivo *er-example-server.c* verifique se as macros "REST_RES_[HELLO e TOGGLE]" possuem valor 1 e as demais macros em 0. Em caso negativo, modifique de acordo. Agora as configurações preliminares estão prontas.

[44]Para mais detalhes sobre os arquivo consulte: https://github.com/contiki-os/contiki/tree/master/examples/er-rest-example

[45]https://addons.mozilla.org/en-US/firefox/addon/copper-270430

Na pasta do exemplo Erbium Rest execute o comando abaixo:

```
make TARGET=cooja server-only.csc
```

Este comando iniciará uma simulação previamente salva. Esta simulação consta 2 dispositivos *Tmotes*, o dispositivo de ID 1 é um roteador de borda e o ID 2 emula um dispositivo executando *er-example-server.c* como *firmeware*.

Em um novo terminal execute o comando abaixo:

```
make connect-router-cooja
```

Como na primeira prática (Seção 1.4.1), este comando executará o programa `tunslip6` e realizará o mesmo procedimento anteriormente citado. Inicie a simulação, abra o Mozilla Firefox e digite: `coap://cooja2:5683/`[46].

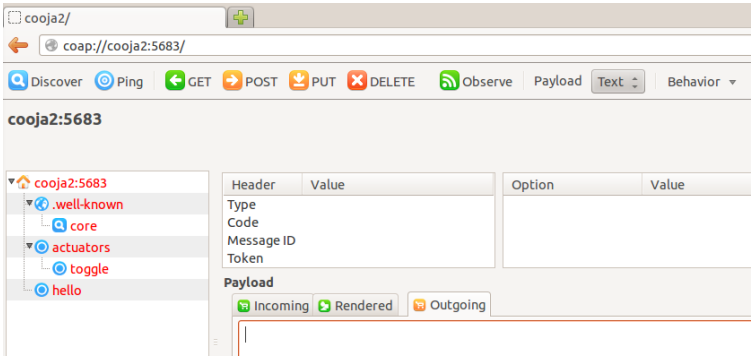


Figura 1.12. Resultado para `coap://cooja2:5683/`

Como resultado o *Mozilla Firefox* deverá abrir o ambiente do Copper. A Figura 1.12 exemplifica a situação atual. No lado esquerdo é possível verificar os recursos disponíveis pelo servidor (cooja2). Para experimentar os recursos providos pelo cooja2, selecione o recurso "hello". Note que a URI foi alterada, em seguida, pressione o botão "GET" do ambiente Copper e observe o resultado. Já no recurso "toggle" pressione o botão "POST" e observe que o LED RED do cooja2 (no simulador) estará ligado, repita o processo e verifique novamente o LED.

É recomendável que os leitores alterem os recursos disponíveis pelo `cooja2` no arquivo *er-example-server.c* modificando os recursos ativos no CoAP server convenientemente. Para fazer isto, comente ou descomente os "`rest_activate_resource()`" presentes no código. No material extra do curso, existem outras simulações e exemplos de uso. Vale ressaltar, que o mote emulado (*Tmote Sky*) apresenta limitações de memória, como a maioria, dos objetos inteligentes e, por esse motivo, nem sempre todos os recursos poderão está ativos ao mesmo tempo. Em caso de excesso de memória, o simulador ou compilador para o dispositivo alertará tal problema.

Comentários. O primeiro exercício utiliza na prática os conceitos aprendidos na seções anteriores. Mostrou-se como funciona uma rede de objetos inteligentes utilizando os protocolos que representam o estado-da-arte para endereçar dispositivos (6LoWPAN) e rotear mensagens (RPL) através de implementações disponíveis no Contiki. A segunda prática mostra como acessar, de modo padronizado, os recursos dos objetos inteligentes por meio

[46]É importante frisar que o CoAP utiliza a porta `5683` como padrão.

do protocolo CoAP, o qual emprega REST e URI para identificar unicamente os recursos disponíveis nos objetos inteligentes. No conteúdo disponível na Web do capítulo[47] existem exemplos para implementação em dispositivos reais.

O conteúdo teórico e prático visto até o momento elucidaram as bases que sustentam o funcionamento da IoT hoje. Os conceitos e práticas são importantes para melhor compreender as próximas seções, pois será assumido que existe uma rede de dispositivos inteligentes funcional e que os recursos providos pelos dispositivos possam ser acessados utilizando, por exemplo, o CoAP ou similares.

1.5. Gerenciamento e Análise de Dados

Uma das principais características da IoT diz respeito à sua capacidade de proporcionar conhecimento sobre o mundo físico, a partir da grande quantidade de dados coletados pelos seus sensores. Por meio da mineração destes dados, é possível descobrir padrões comportamentais do ambiente ou usuários e realizar inferências eles. Por exemplo, é possível concluir, a partir dos dados obtidos, sobre fenômenos naturais, de modo a permitir que aplicações possam antecipar condições meteorológicas e tomar decisões com base nisso. Obviamente, os maiores beneficiados com isto são os próprios usuários, que terão melhorias na qualidade de suas vidas.

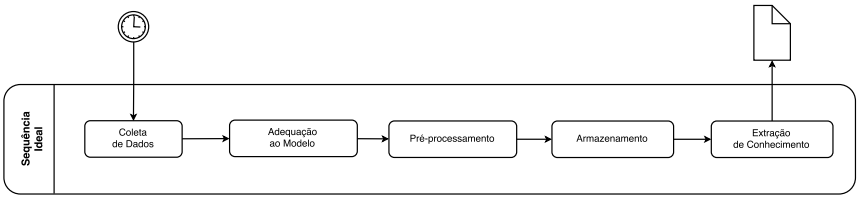
Contudo, para se extrair todo o potencial contido nos dados da IoT, é necessário que, primeiro, algumas medidas sejam tomadas. Nesta seção, destacamos as principais características e desafios encontrados ao se lidar com dados oriundos de sensores. Além disso, apresentamos algumas das principais técnicas utilizadas para modelagem e processamento destes dados, até a extração de conhecimento, para melhoria da qualidade dos serviços em cenários nos quais a IoT é empregada. Por fim, apontamos possíveis direcionamentos para aqueles que desejarem se aprofundar mais no assunto.

1.5.1. Descrição do problema

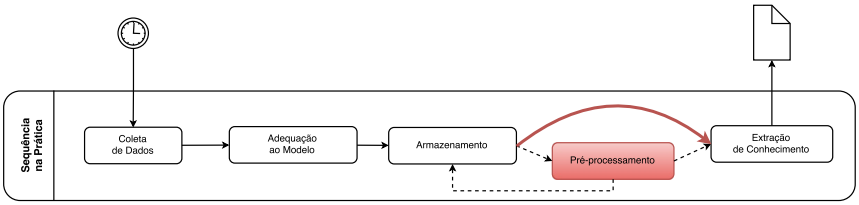
Do ponto de vista dos usuários e suas aplicações, a "*raison d'être*" (razão de ser) da IoT é, justamente, a extração de conhecimento a partir dos dados coletados pelos seus sensores. Extrair um conhecimento refere-se a modelar e analisar dados definindo uma semântica de forma a tomar decisões adequadas para prover um determinado serviço [Barnaghi et al. 2012]. Tomando como exemplo o cenário de *smart grids* [Yan et al. 2013], uma arquitetura de IoT pode auxiliar a controlar e melhorar o serviço de consumo de energia em casas e edifícios. Por meio da IoT, as fornecedoras de energia podem controlar os recursos proporcionalmente ao consumo e possíveis falhas na rede elétrica. Isso acontece por meio das diversas leituras que são coletadas por objetos inteligentes e são analisadas para prevenção e recuperação de falhas, aumentando, assim, a eficiência e qualidade dos serviços.

Para melhor entender o processo de extração de conhecimento, na Figura 1.13a são apresentadas as etapas ideais, que vão desde a coleta dos dados brutos até a extração de conhecimento a partir deles. As etapas de adequação dos dados a um modelo, de pré-processamento e armazenamento são essenciais para que eles estejam aptos a serem

[47] http://homepages.dcc.ufmg.br/~bruno.ps/iot-tp-sbrc-2016/



(a) Representação da sequência ideal



(b) Representação da sequência na prática

Figura 1.13. Etapas para extração de conhecimento a partir de dados de sensores

processados e para que técnicas de inferências possam ser aplicadas sobre eles.

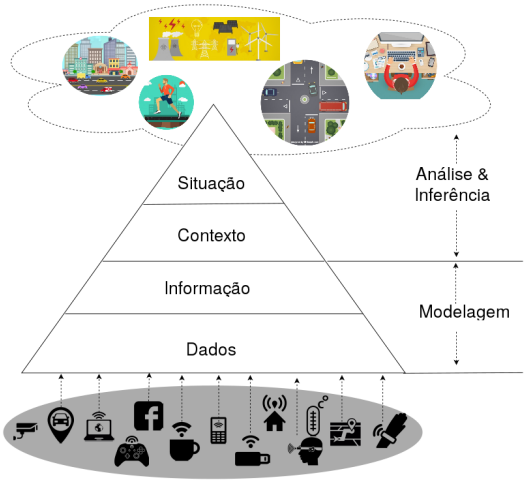


Figura 1.14. Hierarquia dos níveis de conhecimento a partir de dados brutos de sensores

Do ponto de vista do conhecimento em si, uma outra abordagem que pode-se aplicar sobre este processo está ilustrada na Figura 1.14. Nela, a transformação da informação a partir de dados brutos de sensores pode ser entendida por meio de uma hierarquia de níveis de conhecimento. Estes níveis podem ser sub-divididos em dois momentos: (i) modelagem, e (ii) análise e inferência. Na modelagem, cujo principal objetivo é o de adicionar semântica aos dados, depara-se com um grande volume de dados oriundos de diversos tipos de sensores. Uma particularidade desafiadora para manipular esses dados é o fato de serem originados de múltiplas fontes e, em muitos casos, heterogêneas. Nesse sentido, algumas técnicas de pré-processamento, como fusão e representação de dados,

podem ser adotadas, após isso, os dados são armazenados em uma representação adequada. O segundo momento consiste em interpretar tais dados visando obter contexto a partir deles e, com isso, realizar inferências para se extrair conhecimento quanto à situação de um determinado aspecto, seja ele um fenômeno natural ou estado de uma entidade.

Contudo, apesar de cada uma destas etapas ser de fundamental importância para o processo de extração de conhecimento como um todo, na prática não é bem isso o que acontece. Como ilustrado na Figura 1.13b, o que geralmente acontece é a ausência da etapa de pré-processamento dos dados antes que eles sejam armazenados. Os dados coletados dos sensores são simplesmente armazenados para posterior utilização, o que pode comprometer todas as inferências subsequentes. Assim, uma vez de posse dos dados armazenados, o que podemos fazer é realizar as atividades de pré-processamento em uma etapa adicional, antes que os dados sejam utilizados para a extração de conhecimento.

Considerando o cenário representado pelo que ocorre na prática, no restante desta seção aprofundamos nossas considerações sobre cada uma destas etapas.

1.5.2. Modelagem de dados

Dados brutos obtidos pelos sensores dificilmente possuem explicitamente uma organização hierárquica, relacionamentos ou mesmo um formato padrão para manipulação. Esta etapa de modelagem refere-se em definir uma representação para manipular e trabalhar com esses dados visando uma interoperabilidade e formatos padrões interpretáveis. Nesse sentido, aplica-se a modelagem que consiste em definir um conjunto de fatores tais como atributos, características e relações do conjunto de dados sensoriados. Tais fatores variam de acordo com o domínio da aplicação e, consequentemente, uma determinada solução de modelagem se torna mais adequada que outra. A modelagem aqui empregada consiste em representações conceituais de como representar a informação. As representações apresentadas são: *key-value*, *markup scheme*, *graphical*, *object based*, *logic based* e *ontology based modeling*. A aplicabilidade dessas representações pode variar de acordo com o domínio da aplicação. Portanto, cada representação é descrita abaixo considerando suas vantagens e desvantagens em uma perspectiva geral. Um estudo mais aprofundado pode ser encontrado em [Bettini et al. 2010].

Key-value based: nesta representação os dados são modelados como um conjunto de chaves e valores em arquivos de texto. Apesar dessa ser a forma mais simples de representação da informação entre as apresentadas aqui, ela possui muitas desvantagens como a complexidade de organizar e recuperar quando o volume de dados aumenta, além da dificuldade de realizar relacionamentos entre os dados.

Markup scheme based: um *markup scheme* utiliza *tags* para modelar os dados. Linguagens de marcação (e.g., XML[48]) e mecanismos de troca de dados (e.g, JSON[49]) podem ser utilizados para este fim e são bastante aplicados para armazenamento e transferência de dados. A vantagem dessa técnica consiste na estruturação hierárquica dos dados e acesso aos dados utilizando as *tags*. Recentemente, o W3C adotou o EXI [50] o qual consiste de uma representação compacta do XML. EXI pode ser uma relevante alternativa ao XML

[48] https://www.w3.org/XML/
[49] https://tools.ietf.org/html/rfc4627
[50] https://www.w3.org/TR/exi/

para IoT, pois é adequado para aplicações com recursos computacionais limitados. A SensorML[51] (*Sensor Model Language*) provê uma especificação exclusiva para sensores considerando diversos aspectos como localização e metadados.

Object based: esta técnica permite a construção de uma modelagem considerando o relacionamento e a hierarquia entre os dados. Um *Markup scheme* apresenta limitações no sentido que as *tags* são palavras chaves de um domínio e não fornecem uma semântica para a interpretação pela máquina. Técnicas de modelagem como UML (*Unified Modelling Language*) e ORM (*Object Role Modelling*) são preferíveis à *Markup scheme*, pois permitem capturar relacionamentos em um contexto. Esse tipo de modelagem pode ser facilmente vinculada a uma linguagem de propósito geral e, consequentemente, integrável à sistemas de inferência e cientes de contexto. No entanto, não tem a mesma predisposição para realizar inferência das representações baseadas em lógica e ontologia.

Logic based: neste tipo de modelagem, expressões lógicas e regras são usadas para representar a informação. A representação lógica é mais expressiva que as anteriores do ponto de vista que possibilita a geração de novas informações a partir dos dados. Dessa forma, a partir de informações de baixo nível pode-se ter regras para compor informações de alto nível. A desvantagem dessa representação é a dificuldade em generalização das regras, dificultando a reusabilidade e aplicabilidade.

Ontology based: nesta forma de representação, as informações são modeladas como ontologia seguindo os conceitos da *Semantic Web*. Uma ontologia define um conjunto de tipos, propriedades e relacionamentos de entidades dentro de um tema considerando aspectos espaciais e temporais [Jakus et al. 2013]. Para a área de sensores e IoT, surgiu a *Sensor Semantic Web* que refere-se a aplicar o conhecimento da *Web* Semântica contemplando o domínio de sensores. Dessa forma, tecnologias consolidadas podem ser utilizadas, tais como RDF e OWL, para modelar o conhecimento de domínio e estruturar as relações definidas na ontologia. A desvantagem da representação baseada em ontologia concentra-se no consumo de recursos computacionais.

1.5.3. Armazenamento

Para que a grande quantidade de dados gerados pelos sensores da IoT possa ser posteriormente analisada e processada, a etapa de armazenamento faz-se essencial. Como apresentado na Figura 1.6(III), uma das principais formas de acesso a estes dados, e a mais comumente encontrada na prática, acontece por meio de servidores de armazenamento. Muitos destes estão disponíveis sob a forma de plataformas computacionais especificamente voltadas para prover serviços para a IoT. Na Tabela 1.4 apresentamos algumas plataformas para IoT atualmente disponíveis, destacando algumas das suas principais características.

A maioria destas plataformas baseiam suas funcionalidades de acordo com os modelos de dados definidos, assim, logo após coletados, os dados quando adequados ao modelo serão armazenados de forma a possibilitar sua consulta subsequente. Porém, ao contrário do que foi discutido quanto os aspectos de modelagem mais adequados ao domínio de aplicação, o que geralmente ocorre, na prática, é a utilização de um modelo mais

[51] http://www.sensorml.com/index.html

simples e genérico possível, que se adeque ao mais variado número de aplicações. Neste caso, os modelos que se encontram nas principais plataformas são baseados em *key-value* e *markup scheme*. Estes modelos são utilizados para que os usuários possam dar semântica aos seus dados, descrevendo coisas como os tipos dos dados, formatos, etc. Além disso, algumas outras meta informações também podem ser providas, como a localização dos sensores, uma descrição textual do que o sensor representa e algumas *tags* que poderão ser usadas como palavras-chave para consultas.

Além do armazenamento, algumas plataformas também disponibilizam outros serviços, como a marcação de tempo (*timestamp*) de todos os dados recebidos, algumas funcionalidades de processamento, que geralmente são pré-definidos e acessados sob a forma de *wizards* ou *dashboards*, definição de regras para a execução de atividades com base em eventos ou comportamento dos dados, entre outros. Para mais detalhes sobre o projeto e implementação de plataformas para IoT, ver o trabalho de [Paulo F. Pires 2015].

1.5.4. Pré-processamento

Para que os dados possam ser processados adequadamente, eles devem possuir qualidade suficiente para os processos a serem empregados. Apesar de ser difícil de se conceituar, uma forma simples de entender a qualidade aqui discutida se refere aos dados que estão aptos ao uso [Bisdikian et al. 2013]. Nesse sentido, um passo fundamental para contornar possíveis problemas existentes nos dados coletados, de forma a torná-los aptos ao uso, é a de pré-processamento. Contudo, como visto na Figura 1.13b, apesar de sua importância, esta etapa de pré-processamento é muitas vezes inexistente, sendo os dados armazenados com suas imperfeições. Assim, diante do exposto, uma alternativa que se tem é a inserção de sta etapa de pré-processamento logo antes dos dados serem utilizados para a extração de conhecimento.

A seguir, serão discutidos alguns dos principais problemas que ocorrem no dados da IoT e algumas técnicas comumente utilizadas para reduzir seu impacto durante a etapa de extração de conhecimento.

1.5.4.1. Principais problemas dos dados

Para melhor entendimento dos principais problemas que podem ocorrer nos dados provenientes dos sensores de IoT, [Khaleghi et al. 2013] definem uma taxonomia de classificação partindo de problemas básicos relativos aos aspectos dos dados, nos quais os principais são descritos abaixo.

Imperfeição: refere-se aos efeitos causados por alguma imprecisão ou incerteza nas medidas capturadas pelos sensores. As causas destes problemas podem ser várias, desde a problemas nas leituras devido a falhas de hardware ou calibragem dos sensores, até a fatores externos ao sensor, como do seu posicionamento em locais que adicionam ruídos às suas leituras.

Inconsistência: surge principalmente dos seguintes problemas: (i) dados fora de sequência, isto é, em que a ordem em que foram armazenados ou temporalmente demarcados difere da real ordem de ocorrência no mundo físico, (ii) presença de *outliers* nos dados,

Plataforma	Endereço	Descrição	Tipo de conta
AWS IoT	aws.amazon.com/iot	Plataforma da Amazon voltada para empresas.	Possui conta gratuita, mas pede cartão de crédito para confirmar.
Arrayent	arrayent.com	Plataforma com foco empresarial.	Não disponibiliza conta gratuita.
Axeda	axeda.com	Plataforma com foco empresarial. Provê serviços de gerenciamento e comunicação entre dispositivos.	Não disponibiliza conta gratuita.
Beebotte	beebotte.com	Plataforma com interessantes recursos, incluindo a possibilidade de criação de dispositivos públicos.	Possui conta gratuita, mas com limite de 3 meses de histórico.
BugLabs	dweet.io	Permite *publish-subscribe* para disponibilização dos dados e integração entre sensores.	Possui conta gratuita, mas os dados são apagados após 24h.
Carriots	carriots.com	Plataforma com foco empresarial. Provê serviços de gerenciamento de e comunicação entre dispositivos	Possui conta gratuita, mas com várias limitações, *e.g.*, número de dispositivos e acessos.
Electric imp	electricimp.com	Plataforma em nuvem que integra o conjunto de soluções dos dispositivos *electric imp*.	Possui conta gratuita.
Evrythng	evrythng.com	Plataforma com foco empresarial. Permite *publish/subscribe* para disponibilização dos dados de sensores.	Disponibiliza conta gratuita para desenvolvedor.
Exosite	exosite.com	Plataforma com foco empresarial.	Possui conta gratuita, mas limitada.
Flowthings	flowthings.io	Plataforma com interessantes conceitos para a integração de sensores.	Possui conta gratuita para desenvolvedor.
IBM Bluemix	bluemix.net	Plataforma da IBM com foco empresarial.	Possui conta gratuita, mas pede cartão de crédito para confirmar.
Kaa	kaaproject.org	*Middleware open source* disponível para a criação de sua própria plataforma para IoT.	Gratuita.
Lelylan	lelylan.com	Projeto *open source* com interessantes conceitos de arquitetura que pode ser utilizado para a criação de sua própria plataforma.	Gratuita.
Linkafy	linkafy.com	Plataforma voltada para o controle de dispositivos residenciais.	Possui conta gratuita, mas limitada a apenas dispositivo.
mbed	mbed.com	Plataforma que integra o conjunto de soluções que suportam os dispositivos mbed da ARM.	Possui conta gratuita, com limitações.
Microsoft Azure IoT	microsoft.com/iot	Plataforma da Microsoft voltada a IoT com foco empresarial.	Possui conta gratuita de um mês para testes.
Open.sen.se	open.sen.se	Plataforma interessante para integração dos sensores e seus dados.	Conta apenas após requisição.
OpenSensors	opensensors.io	Plataforma robusta com o foco principal para sensores abertos. Permite *publish-subscribe* para disponibilização dos dados de sensores.	Conta gratuita disponível, paga-se apenas para ter sensores privados.
PubNub	pubnub.com	Plataforma robusta com diversas funcionalidades voltadas para IoT.	Possui conta gratuita com limitações.
SensorCloud	sensorcloud.com	Plataforma com foco empresarial.	Possui conta gratuita, mas limitada.
SensorFlare	sensorflare.com	Plataforma para integração de sensores, mas apenas suporta alguns fabricantes e modelos.	Possui conta gratuita.
Sentilo	sentilo.io	Projeto gratuito e disponível para a criação de sua própria plataforma. Voltada às arquiteturas de *smart cities*.	Gratuita.
Shiftr	shiftr.io	Interessantes conceitos para a integração de sensores de forma intuitiva.	Possui conta gratuita.
ThingPlus	thingplus.net	Plataforma sul koreana com o foco interessante sistema de definição de regras para integração entre sensores.	Possui conta gratuita, mas limitada.
ThingSquare	thingsquare.com	Plataforma voltada ao controle de dispositivos e integração via celular.	Possui conta de desenvolvedor gratuita.
ThingSpeak	thingspeak.com	Plataforma robusta, com várias funcionalidades, como sensores públicos e busca por histórico.	Conta gratuita disponível.
ThingWorx	thingworx.com	Plataforma com recursos interessantes, mas mais voltada a soluções empresariais.	Conta gratuita, mas com limitações.
Xively	xively.com	Plataforma robusta, disponibiliza várias funcionalidades como busca por sensores públicos e histórico.	Foco empresarial, com conta paga disponível e gratuita apenas via solicitação.

Tabela 1.4. Algumas das principais plataformas para IoT

observações que estão bem distantes das demais observações realizadas, causadas por alguma situação inesperada, e (iii) dados conflitantes, quando diferentes sensores medindo um mesmo fenômeno geram dados diferentes, agregando uma dúvida sobre qual sensor seria mais confiável.

Discrepância: este é um problema causado, principalmente, quando diferentes tipos de sensores são utilizados para coletar dados sobre um mesmo fenômeno. Por exemplo, sensores físicos coletando informações sobre o tráfego, comparados com câmeras de monitoramento, ou ainda, sensores sociais, que coletam informações dadas por usuários em seus aplicativos móveis [Silva et al. 2014].

Além destes problemas, também podemos destacar outros problemas dos dados, principalmente no cenário atual da IoT, onde usuários comuns também estão participando de forma colaborativa da geração destes dados [Borges Neto et al. 2015]. Devido à popularização e redução dos custos dos sistemas computacionais embarcados, muitos usuários estão criando seus próprios sensores, seguindo um modelo DIY (*Do It Yourself*). Neste cenário, eles são responsáveis pela implantação, coleta e distribuição dos dados dos sensores, geralmente tornando-os públicos através das principais plataformas de IoT. Contudo, por se tratarem de sensores "particulares", não há nenhuma garantia quanto à qualidade dos dados gerados, nem mesmo da disponibilidade dos sensores.

Assim, alguns problemas que podem surgir neste novo cenário são: (i) ausência de descrição dos dados gerados, quando os usuários submetem os dados coletados pelos seus sensores para uma plataforma sem descrever de que se trata o dado, dificultando qualquer posterior utilização deste dado por outros usuários, (ii) disponibilidade dos sensores, quando os sensores param de funcionar, temporária ou permanentemente, sem mais detalhes se voltaram a operar ou não, geralmente de acordo com as necessidades dos próprios usuários, (iii) imprecisão das leituras causadas pelo baixo custo e qualidade dos sensores "caseiros", geralmente utilizando-se de técnicas alternativas para medir um fenômeno, as leituras destes sensores dos usuários podem diferir em magnitude dos dados coletados por sensores mais profissionais quanto a um mesmo fenômeno.

Para exemplificar isto, a Figura 1.15 apresenta duas medições distintas para um mesmo fenômeno, a velocidade do vento, na região da Espanha, realizadas por um sensor público, disponibilizado por um usuário através da plataforma *ThingSpeak*[52] e pelo serviço meteorológico do *Weather Underground*[53], no período entre 12 de setembro a 31 de outubro de 2015. Observa-se que embora bem relacionadas, com um coeficiente de correlação equivalente a 0.915 entre eles, suas magnitudes são diferentes. Além disso, também pode-se notar a incompletude destes dados, quando há lacunas devido à falta de dados coletados pelos sensores.

1.5.4.2. Principais técnicas de pré-processamento

O pré-processamento consiste na aplicação de técnicas, em sua maioria estatísticas e demais operações matemáticas, sobre os dados com o intuito melhorar a sua qualidade,

[52] https://thingspeak.com
[53] https://www.wunderground.com

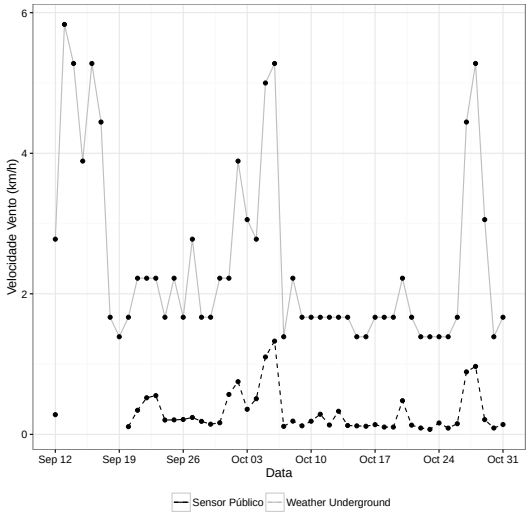


Figura 1.15. Velocidade do Vento (km/h) coletada na região de Madri, Espanha, medida por um sensor público, disponível na plataforma *ThingSpeak* (linha tracejada) e pelo serviço meteorológico do *Weather Underground* (linha sólida)

contornando possíveis problemas existentes e removendo imperfeições. A complexidade desta etapa pode ser alta, principalmente neste cenário em questão, quando os dados são heterogêneos, oriundos de diversas fontes e com diferentes problemas. Nesse sentido, a decisão sobre qual técnica aplicar varia de acordo com o problema encontrado nos dados e, além disso, do que se espera fazer com estes dados. Assim, a seguir será apresentada uma visão geral sobre algumas das técnicas comumente utilizadas para alguns dos problemas citados, e apontamos algumas referências, para que leitores mais interessados possam se aprofundar no assunto.

Imprecisões e outliers: a presença de imprecisões e *outliers* em dados é um problema muito comum em diversas áreas e um ponto bastante estudado pela comunidade científica. As soluções mais comumente utilizadas para contornar estes problemas são baseadas em técnicas de suavização dos dados. Por exemplo, (i) a utilização de médias móveis (*moving average*), onde as amostras são suavizadas por meio da média de amostras vizinhas, e (ii) de interpolação (polinomial ou *splines*), onde são definidas funções, *e.g.*, polinomiais, que melhor se ajustam às amostras contidas nos dados. Com isto, é possível diminuir o efeito dos ruídos causados pelas imperfeições nos dados. Para mais detalhes, uma consulta pode ser [Morettin and Toloi 2006].

Lacunas nos dados: o caso de lacunas nos dados acontece por fatores diversos e que podem causar diferentes níveis de prejuízo em sua extração de conhecimento. Um caso mais simples seria a lacuna causada por uma falha esporádica na operação dos sensores, por razões não associadas ao fenômeno monitorado, causando uma lacuna aleatória ou MAR (*missing at random*). Este tipo de falha pode ser contornada de forma mais fácil, por exemplo, por meio de interpolação para completar os dados faltantes. Mas, dependendo da forma como isso acontece, podem ser inseridos lacunas de forma sistemáticas nos

dados, prejudicando a sua posterior análise. Por exemplo, um sensor que pára de coletar dados quando a temperatura atinge certos níveis de valores, ou quando um usuário desliga o sensor à noite quando deixa o escritório. Este tipo de lacunas que não são geradas aleatoriamente, ou MNAR (*missing not at random*) são mais problemáticas para o seu processamento. Outras técnicas também podem ser aplicadas, desde a simples remoção do período contendo lacunas até a imputação dos dados faltantes por meio da estimativa a partir dos demais dados existentes. Mais detalhes em [Schafer and Graham 2002].

Diferença de granularidade: devido à grade quantidade de sensores heterogêneos, um problema relevante que surge diz respeito às diferenças de amostragem que cada sensor possui. Isso pode causar diferenças significativas na granularidade dos dados de diferentes sensores. Por exemplo, para um mesmo fenômeno em um mesmo intervalo de tempo, podemos ter um sensor coletando dados a cada 5 segundos e outro a cada 1 minuto. Para que seja possível combinar os dados destes sensores, um primeiro pré-processamento seria igualar a quantidade de amostras de cada um deles. Para isto, algumas técnicas podem ser utilizadas e, uma simples mas eficaz é a PAA (*Piecewise Aggregate Approximation*), que é empregada pelo algoritmo de clusterização SAX (*Symbolic Aggregate approXimation*) [Lin et al. 2003], para a redução da dimensão do conjunto de dados por meio da agregação de dados, similar ao que é feito com médias móveis. Assim, pode-se transformar os dados de forma a possuírem a mesma dimensão. Ainda sobre a redução de dimensão, esta é uma técnica bem válida para o cenário de IoT, pois espera-se uma grande quantidade de dados sendo gerados. Outras estratégias também podem ser adotadas, como a transformação via *wavelets* ou Fourier, mais detalhes em [Rani and Sikka 2012, Warren Liao 2005].

Combinação de diversas fontes: para a combinação de dados de diversas fontes, como é o caso da IoT, técnicas de fusão de dados podem ser aplicadas de forma a compor uma única representação destes dados. Fusão de dados é o método de combinar dados de diversos sensores para produzir uma semântica mais precisa e muitas vezes inviável de ser obtida a partir de um único sensor. A utilização de fusão de dados nesta etapa de pré-processamento tem o intuito principal de cobrir os problemas dos dados de um sensor através dos dados coletados por outros sensores similares, gerando assim um novo dado combinado mais preciso. Dessa forma, são aplicados métodos automáticos ou semi-automáticos para transformar dados de diferentes fontes em uma representação que seja significativa para a tomada de decisão e inferência. Em cenários de IoT nos quais existem milhares de sensores a fusão de dados é uma poderosa ferramenta tanto processo de extração de conhecimento quanto para reduzir recursos computacionais semelhante ao que ocorre em redes de sensores sem fio. Várias técnicas de fusão podem ser aplicadas, como, por exemplo, filtros de Kalman [Khaleghi et al. 2013]. Outra técnicas muito utilizadas para a combinação de sensores distintos se baseiam na clusterização de dados similares, com base em alguma métrica de distância, mas mais detalhes sobre isso algumas técnicas de clusterização vide [Rani and Sikka 2012, Warren Liao 2005].

1.5.5. Extração de conhecimento

Como visto na Figura 1.14, a modelagem consiste em explorar os dados brutos para estruturá-los em uma específica representação. Aumentando o nível de abstração na hierarquia, contexto refere-se a qualquer informação que pode ser utilizada para caracterizar a situação de uma entidade. Sendo essa entidade um objeto, lugar ou pessoa que é con-

siderada relevante para interação entre um usuário e uma aplicação, incluindo o usuário e a própria aplicação [Dey 2001]. Enquanto que uma situação representa a interpretação semântica de contextos, geralmente derivado pela combinação de diversas informações contextuais, proporcionando conhecimento sobre o mundo físico [Dobson and Ye 2006]. Particularmente, a etapa de análise e inferência busca definir o contexto e a situação a partir dos dados coletados por sensores.

Para exemplificar um cenário, Bettini [Bettini et al. 2010] especifica uma situação "reunião está ocorrendo" considerando as seguintes informações contextuais: localização de várias pessoas e agenda de atividades delas, informação de sensores nos pisos, quais os dispositivos ativos na sala, o som ambiente e as imagens de câmeras. Ao longo do tempo essas informações contextuais vão se alterando, mas a situação ainda é a mesma. Além disso, outras informações contextuais poderão ser incorporadas, enquanto essas utilizadas podem se tornar obsoletas. Dessa forma, o objetivo da inferência refere-se a estratégia de extrair um conhecimento (i.e., semântica de uma situação) baseado nos dados coletados. Esta etapa é relacionada com a etapa de modelagem, pois algumas técnicas de modelagem são preferíveis de serem utilizadas com determinado tipo de técnica de inferência. A seguir, são apresentadas algumas categorias de técnicas úteis para inferência de situações. Uma revisão mais abrangente sobre tais técnicas pode ser encontrada em [Bettini et al. 2010, Perera et al. 2014].

Aprendizagem Supervisionada: Nesse tipo de técnica, uma primeira coleção de dados é coletada para compor o conjunto de treinamento. Para tanto, esse dados devem ser rotulados em classes específicas [Mitchell 1997] [Bishop 2006]. Em seguida, cria-se um modelo (ou função) que aprende a classificar dados de acordo com os dados que foram utilizados na etapa de treinamento. As árvores de decisão, redes neurais artificiais baseadas neste tipo de aprendizado e máquinas de vetores de suporte são exemplos de técnicas de aprendizagem supervisionada. Dois fatores importantes são a seleção de características significativas e uma considerável quantidade de dados para classificação.

Aprendizagem não supervisionada: Nesta categoria de técnicas, não existe nenhum conhecimento a priori sobre que padrões ocorrem nos dados e o objetivo é encontrar um modelo (ou função) que descubra padrões implícitos em um conjunto de dados não rotulados [Mitchell 1997] [Bishop 2006]. Dessa forma, é difícil uma confiável validação dos resultados obtidos e os resultados geralmente não são previsíveis. Exemplos clássicos desta categoria são as técnicas de clusterização tais como K-Means e clusterização hierárquica que agrupam os elementos se baseando em métricas de similaridade entre eles.

Regras: Este tipo de técnica consiste em definir regras condicionais. Apesar de ser a forma mais simples de inferência, ela apresenta dificuldades de generalização e pouca extensibilidade para diferentes aplicações.

Ontologias: Nessa categoria, ontologias podem ser utilizadas tanto na modelagem como para a inferência [Min et al. 2011]. Assim, já se tem a vantagem de modelar um conhecimento sabendo o domínio da aplicação e o mapeamento das possibilidades de inferência. No entanto, apresenta a desvantagem de não tratar ambiguidades ou resultados inesperados. Essas desvantagens podem ser contornadas fazendo a combinação com regras.

Lógica probabilística: Também conhecida como inferência probabilística refere-se a utilizar a teoria de probabilidade para lidar com as incertezas existentes em um processo dedutivo [Murphy 2012]. Ao contrário das ontologias, esta categoria de técnicas pode lidar com ambiguidades considerando as probabilidades associadas para realizar a inferência. A desvantagem consiste em descobrir tais probabilidades. *Hidden Markov Models* (HMM) é um exemplo clássico e tem sido aplicado em cenários de reconhecimento de atividades de pessoas.

1.6. Considerações Finais

No decorrer do capítulo, o leitor foi apresentado a diversos aspectos da Internet das Coisas, passando por questões tanto teóricas quanto práticas. Por ser uma área extensa, ainda existem pontos que não foram discutidos ao longo do texto e, por esse motivo, alguns desses assuntos serão brevemente referenciados aqui e no conteúdo online[54] como leitura complementar.

A IoT, neste momento, passa por questões que dizem respeito a sua forma, proprietários e regulamentações. Neste sentido, os próximos anos serão fundamentais para as definições e padronizações tecnológicas para IoT. Neste sentido, empresas como Google, Apple e outras estão lançando seus próprios ecossistemas IoT, respectivamente, *Google Weave* e *Apple HomeKit*. Entretanto, cada um possui protocolos, tecnologias de comunicação e padrões particulares. Isto torna a IoT não padronizada o que leva a um ecossistema que pode não prosperar. Assim, é preciso que a comunidade acadêmica e empresarial atentem-se às padronizações e na construção de um ecossistema favorável à IoT. Com isto será possível tornar os dispositivos *Plug & Play* e tornar a IoT livre de padrões proprietários. Em [Ishaq et al. 2013], os autores realizam uma revisão geral das padronizações em IoT.

Outra questão altamente relevante é a segurança para IoT. Este ponto, é uma das principais barreiras que impedem a efetiva adoção da IoT, pois os usuários estão preocupados com a violação dos seus dados. Deste modo, a segurança desempenha papel chave para a adoção da IoT. O escritor Dominique Guinard resume, em seu artigo sobre políticas para IoT[55], que "*A segurança dos objetos inteligentes é tão forte quanto seu enlace mais fraco*", isto é, as soluções de segurança ainda não estão consolidadas, portanto soluções devem ser propostas e discutidas detalhadamente.

Neste trabalho, foram descritos alguns dos princípios básicos da IoT através de uma abordagem teórica e prática. O aspecto dos objetos inteligentes e as tecnologias de comunicação foram abordados primeiramente. De forma complementar, discutimos os softwares que orquestram o funcionamento da IoT. Foram realizadas duas atividades práticas para consolidar o conteúdo. Também apresentou-se o modo como gerenciar e analisar dados oriundos de dispositivos inteligentes, destacando as principais técnicas utilizadas.

Por fim, um das maiores oportunidades, tanto para a academia quanto para a indústria, é o projeto de aplicações e serviços que gerem valor a partir dos dados obtidos

[54] http://homepages.dcc.ufmg.br/~bruno.ps/iot-tp-sbrc-2016/
[55] http://techcrunch.com/2016/02/25/the-politics-of-the-internet-of-things/

por dispositivos inteligentes. Ou seja, não é o fato de termos dados em uma grande quantidade que isso implique em um valor. Pelo contrário, é necessário investir fortemente em pesquisa para gerarmos produtos que sejam úteis para a sociedade.

Referências

[Al-Fuqaha et al. 2015] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *Communications Surveys & Tutorials, IEEE*, 17(4):2347–2376.

[Angell et al. 1983] Angell, J. B., Barth, P. W., and Terry, S. C. (1983). Silicon micromechanical devices. *Scientific American*, 248:44–55.

[Ashton 2009] Ashton, K. (2009). That 'internet of things' thing. *RFiD Journal*, 22(7):97–114.

[Bagula and Erasmus 2015] Bagula, B. and Erasmus, Z. (2015). Iot emulation with cooja. ICTP-IoT Workshop.

[Barnaghi et al. 2012] Barnaghi, P., Wang, W., Henson, C., and Taylor, K. (2012). Semantics for the Internet of Things. *International Journal on Semantic Web and Information Systems*, 8(1):1–21.

[Bettini et al. 2010] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D. (2010). A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180.

[Bisdikian et al. 2013] Bisdikian, C., Kaplan, L. M., and Srivastava, M. B. (2013). On the quality and value of information in sensor networks. *ACM Transactions on Sensor Networks*, 9(4):1–26.

[Bishop 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Borges Neto et al. 2015] Borges Neto, J. B., Silva, T. H., Assunção, R. M., Mini, R. A. F., and Loureiro, A. A. F. (2015). Sensing in the Collaborative Internet of Things. *Sensors*, 15(3):6607–6632.

[Boulis et al. 2011] Boulis, A. et al. (2011). Castalia: A simulator for wireless sensor networks and body area networks. *NICTA: National ICT Australia*.

[Chaouchi 2013] Chaouchi, H. (2013). *The internet of things: connecting objects*. John Wiley & Sons.

[Chlipala et al. 2004] Chlipala, A., Hui, J., and Tolle, G. (2004). Deluge: data dissemination for network reprogramming at scale. *University of California, Berkeley, Tech. Rep*.

[Clausen et al. 2013] Clausen, T., Yi, J., Herberg, U., and Igarashi, Y. (2013). Observations of rpl: Ipv6 routing protocol for low power and lossy networks. draft-clausen-lln-rpl-experiences-05.

[Da Xu et al. 2014] Da Xu, L., He, W., and Li, S. (2014). Internet of Things in industries: A survey. *Industrial Informatics, IEEE Transactions on*, 10(4):2233–2243.

[Dey 2001] Dey, A. K. (2001). Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7.

[Dobson and Ye 2006] Dobson, S. and Ye, J. (2006). Using fibrations for situation identification. In *Pervasive 2006 workshop proceedings*, pages 645–651.

[Doddavenkatappa et al. 2012] Doddavenkatappa, M., Chan, M. C., and Ananda, A. L. (2012). *Testbeds and Research Infrastructure. Development of Networks and Communities: 7th International ICST Conference,TridentCom 2011, Shanghai, China, April 17-19, 2011, Revised Selected Papers*, chapter Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed, pages 302–316. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Dunkels et al. 2004] Dunkels, A., Gronvall, B., and Voigt, T. (2004). Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, LCN '04, pages 455–462, Washington, DC, USA. IEEE Computer Society.

[Fasolo et al. 2007] Fasolo, E., Rossi, M., Widmer, J., and Zorzi, M. (2007). In-network aggregation techniques for wireless sensor networks: a survey. *Wireless Communications, IEEE*, 14(2):70–87.

[Forbes 2014] Forbes (2014). Internet of Things By The Numbers: Market Estimates And Forecasts.

[Gartner 2015] Gartner, I. (2015). Gartner's 2015 Hype Cycle for Emerging Technologies Identifies the Computing Innovations That Organizations Should Monitor.

[Gnawali et al. 2009] Gnawali, O., Fonseca, R., Jamieson, K., Moss, D., and Levis, P. (2009). Collection Tree Protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*.

[Goldstein et al. 2005] Goldstein, S. C., Campbell, J. D., and Mowry, T. C. (2005). Programmable matter. *Computer*, 38(6):99–101.

[Hui 2012] Hui, J. W. (2012). The routing protocol for low-power and lossy networks (rpl) option for carrying rpl information in data-plane datagrams.

[Ishaq et al. 2013] Ishaq, I., Carels, D., Teklemariam, G. K., Hoebeke, J., Abeele, F. V. d., Poorter, E. D., Moerman, I., and Demeester, P. (2013). Ietf standardization in the field of the internet of things (iot): a survey. *Journal of Sensor and Actuator Networks*, 2(2):235–287.

[Jakus et al. 2013] Jakus, G., Milutinovic, V., Omerovic, S., and Tomazic, S. (2013). *Concepts, Ontologies, and Knowledge Representation*. Springer Publishing Company, Incorporated.

[Javaid et al. 2009] Javaid, N., Javaid, A., Khan, I., and Djouani, K. (2009). Performance study of ETX based wireless routing metrics. In *2nd IC4 2009*.

[Kelly et al. 2013] Kelly, S. D. T., Suryadevara, N. K., and Mukhopadhyay, S. C. (2013). Towards the implementation of IoT for environmental condition monitoring in homes. *Sensors Journal, IEEE*, 13(10):3846–3853.

[Khaleghi et al. 2013] Khaleghi, B., Khamis, A., Karray, F. O., and Razavi, S. N. (2013). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44.

[Kovatsch et al. 2011] Kovatsch, M., Duquennoy, S., and Dunkels, A. (2011). A low-power coap for contiki. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 855–860. IEEE.

[Kurose and Ross 2012] Kurose, J. F. and Ross, K. W. (2012). *Computer Networking: A Top-Down Approach (6th Edition)*. Pearson, 6th edition.

[Kushalnagar et al. 2007] Kushalnagar, N., Montenegro, G., and Schumacher, C. (2007). Ipv6 over low-power wireless personal area networks (6lowpans): overview, assumptions, problem statement, and goals. Technical report.

[Levis and Lee 2010] Levis, P. and Lee, N. (2010). TOSSIM: A Simulator for TinyOS Networks.

[Levis et al. 2003] Levis, P., Lee, N., Welsh, M., and Culler, D. (2003). Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 126–137, New York, NY, USA. ACM.

[Levis et al. 2005] Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., and Culler, D. (2005). *Ambient Intelligence*, chapter TinyOS: An Operating System for Sensor Networks, pages 115–148. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Levis et al. 2004] Levis, P., Patel, N., Culler, D., and Shenker, S. (2004). Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, pages 2–2, Berkeley, CA, USA. USENIX Association.

[Lin et al. 2003] Lin, J., Keogh, E., Lonardi, S., and Chiu, B. (2003). A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery - DMKD '03*, pages 2 – 11, New York, New York, USA. ACM Press.

[Liu et al. 2013] Liu, V., Parks, A., Talla, V., Gollakota, S., Wetherall, D., and Smith, J. R. (2013). Ambient backscatter: wireless communication out of thin air. *ACM SIGCOMM Computer Communication Review*, 43(4):39–50.

[Loureiro et al. 2003] Loureiro, A. A., Nogueira, J. M. S., Ruiz, L. B., Mini, R. A. d. F., Nakamura, E. F., and Figueiredo, C. M. S. (2003). Redes de Sensores Sem Fio. In *Simpósio Brasileiro de Redes de Computadores (SBRC)*, pages 179–226.

[Mattern and Floerkemeier 2010] Mattern, F. and Floerkemeier, C. (2010). From the internet of computers to the internet of things. In *From active data management to event-based systems and more*, pages 242–259. Springer.

[Min et al. 2011] Min, Z., Bei, W., Chunyuan, G., and Zhao qian, S. (2011). *Applied Informatics and Communication: International Conference, ICAIC 2011, Xi'an, China, August 20-21, 2011, Proceedings, Part IV*, chapter Application Study of Precision Agriculture Based on Ontology in the Internet of Things Environment, pages 374–380. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Mitchell 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

[Morettin and Toloi 2006] Morettin, P. A. and Toloi, C. M. C. (2006). *Análise de Séries Temporais*. Blucher, São Paulo, 2nd edition.

[Murphy 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

[Österlind] Österlind, F. *A Sensor Network Simulator for the Contiki OS*.

[Patel and Rutvij H. 2015] Patel, K. N. and Rutvij H., J. (2015). A survey on emulation testbeds for mobile ad-hoc networks. *Procedia Computer Science*, 45:581–591.

[Paulo F. Pires 2015] Paulo F. Pires, Flavia C. Delicato, T. B. (2015). Plataformas para a Internet das Coisas.

[Perera et al. 2014] Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2014). Context aware computing for the internet of things: A survey. *Communications Surveys & Tutorials, IEEE*, 16(1):414–454.

[Peterson and Davie 2011] Peterson, L. L. and Davie, B. S. (2011). *Computer Networks, Fifth Edition: A Systems Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition.

[Ramos et al. 2012] Ramos, H. S., Oliveira, E. M., Boukerche, A., Frery, A. C., and Loureiro, A. A. (2012). Characterization and mitigation of the energy hole problem of many-to-one communication in wireless sensor networks. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*, pages 954–958. IEEE.

[Rani and Sikka 2012] Rani, S. and Sikka, G. (2012). Recent Techniques of Clustering of Time Series Data: A Survey. *International Journal of Computer Applications*, 52(15):1–9.

[RERUM 2015] RERUM (2015). Advanced techniques to increase the lifetime of smart objects and ensure low power network operation. *RERUM*.

[Ruiz et al. 2004] Ruiz, L. B., Correia, L. H. A., Vieira, L. F. M., Macedo, D. F., Nakamura, E. F., Figueiredo, C. M., Vieira, M. A. M., Bechelane, E. H., Camara, D., Loureiro, A. A., et al. (2004). Arquiteturas para redes de sensores sem fio.

[Saltzer et al. 1984] Saltzer, J. H., Reed, D. P., and Clark, D. D. (1984). End-to-end arguments in system design. *ACM Transactions on Computer Systems (TOCS)*, 2(4):277–288.

[Santos et al. 2015a] Santos, B., Vieira, M., and Vieira, L. (2015a). eXtend collection tree protocol. In *Wireless Communications and Networking Conference (WCNC), 2015 IEEE*, pages 1512–1517.

[Santos et al. 2015b] Santos, B. P., Menezes Vieira, L. F., and Menezes Vieira, M. A. (2015b). CRAL: a centrality-based and energy efficient collection protocol for low power and lossy networks. In *Computer Networks and Distributed Systems (SBRC), 2015 XXXIII Brazilian Symposium on*, pages 159–170. IEEE.

[Schafer and Graham 2002] Schafer, J. L. and Graham, J. W. (2002). Missing data: Our view of the state of the art. *Psychological Methods*, 7(2):147–177.

[Shelby and Bormann 2011] Shelby, Z. and Bormann, C. (2011). *6LoWPAN: The wireless embedded Internet*, volume 43. John Wiley & Sons.

[Silva et al. 2014] Silva, T., Vaz De Melo, P., Almeida, J., and Loureiro, A. (2014). Large-scale study of city dynamics and urban social behavior using participatory sensing. *Wireless Communications, IEEE*, 21(1):42–51.

[Sundmaeker et al. 2010] Sundmaeker, H., Guillemin, P., Friess, P., and Woelfflé, S. (2010). *Vision and challenges for realising the Internet of Things*, volume 20. EUR-OP.

[Tanenbaum 2002] Tanenbaum, A. (2002). *Computer Networks*. Prentice Hall Professional Technical Reference, 4th edition.

[Tanenbaum 2011] Tanenbaum, A. (2011). *Computer Networks*. Prentice Hall Professional Technical Reference, 5th edition.

[Tsvetko 2011] Tsvetko, T. (2011). Rpl: Ipv6 routing protocol for low power and lossy networks. In *Seminar Sensorknoten: Betrieb, Netze und Anwendungen SS*.

[Varga et al. 2001] Varga, A. et al. (2001). The omnet++ discrete event simulation system. In *Proceedings of the European simulation multiconference (ESM'2001)*, volume 9, page 65. sn.

[Varga and Hornig 2008] Varga, A. and Hornig, R. (2008). An Overview of the OMNeT++ Simulation Environment. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Vasseur et al. 2011] Vasseur, J., Agarwal, N., Hui, J., Shelby, Z., Bertrand, P., and Chauvenet, C. (2011). Rpl: The ip routing protocol designed for low power and lossy networks. Internet Protocol for Smart Objects (IPSO) Alliance.

[Vasseur and Dunkels 2010] Vasseur, J.-P. and Dunkels, A. (2010). *Interconnecting Smart Objects with IP: The Next Internet*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Wang et al. 2015] Wang, F., Hu, L., Zhou, J., and Zhao, K. (2015). A Survey from the Perspective of Evolutionary Process in the Internet of Things. *International Journal of Distributed Sensor Networks*, 2015.

[Warren Liao 2005] Warren Liao, T. (2005). Clustering of time series data - A survey. *Pattern Recognition*, 38(11):1857–1874.

[Weingärtner et al. 2009] Weingärtner, E., Vom Lehn, H., and Wehrle, K. (2009). A performance comparison of recent network simulators. In *Proceedings of the 2009 IEEE International Conference on Communications*, ICC'09, pages 1287–1291, Piscataway, NJ, USA. IEEE Press.

[Yan et al. 2013] Yan, Y., Qian, Y., Sharif, H., and Tipper, D. (2013). A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges. *IEEE Communications Surveys & Tutorials*, 15(1):5–20.

Capítulo

2

Computação Urbana: Tecnologias e Aplicações para Cidades Inteligentes

Carlos Kamienski, Gabriela Oliveira Biondi, Fabrizio Ferreira Borelli, Alexandre Heideker, Juliano Ratusznei, João Henrique Kleinschmidt

Abstract

Cities are becoming smarter as new and existing technologies are increasingly more used for providing a variety of services and applications, for both citizens and municipalities. Smart Cities involve not only technology, but also people and government, working together for equalizing investments in infrastructure and in social and human capital that lead citizens to a higher engagement and participation in city governance processes. The use of Information and Communication Technologies in Smart Cities gives birth to a new area, called Urban Computing, which can be understood as the computational support for building urban environments that offer advanced services to citizens. The traditional ideal of smart cities, coupled with the new developments in areas such as Internet of Things, network softwarization, mobile computing and Big Data may be made possible by urban computing for building smarter societies.

Resumo

Cidades estão se tornando mais inteligentes à medida em que tecnologias novas e existentes são cada vez mais empregadas na oferta de uma grande variedade de serviços e aplicações, tanto para o cidadão, quanto para a própria municipalidade. Cidades inteligentes envolvem não apenas a tecnologia, mas também as pessoas e o governo, unidos para equalizar os investimentos em infraestrutura com aqueles em capital humano e social que levam o cidadão a uma maior participação nos processos de governança. A aplicação das Tecnologias da Informação e da Comunicação em Cidades Inteligentes gera uma nova área, a Computação Urbana, que pode ser compreendida como o suporte computacional para a construção de ambientes urbanos que oferecem serviços avançados aos cidadãos. O ideal já tradicional de cidades inteligentes acoplado aos novos

desenvolvimentos nas áreas de Internet das Coisas, softwarização de redes, computação móvel e Big Data pode ser viabilizado pela computação urbana para criar sociedades mais inteligentes.

2.1. Introdução

A população mundial está fazendo a transição de regiões rurais para centros urbanos e este fato tem grandes implicações na maneira como as cidades são estruturadas e gerenciadas. No mundo inteiro, mais de 50% da população mora em cidades e este número deve chegar a 65% em 2040 [Postscapes 2015]. O Brasil já fez esta transição há algumas décadas e atualmente de acordo com o IBGE, o Senso 2010 identificou que a população urbana contabilizava 84% do total [IBGE 2010]. Atualmente, os 600 maiores centros urbanos geram 60% do PIB global e consomem até 80% de toda a energia. Esse fato aumenta o desafio e a necessidade de construir cidades mais adequadas a prover a infraestrutura e os serviços aos cidadãos.

O ideal de cidades inteligentes é antigo mas somente recentemente as tecnologias de informação e comunicação atingiram um estágio de desenvolvimento que permite que elas se tornem realidade. Existe uma necessidade urgente de que as cidades sejam mais inteligentes na forma de gerenciar a infraestrutura e recursos e serviços oferecidos ao cidadão [Naphade *et al.* 2011]. Para reduzir custos, aumentar a sua eficiência e gerar maior qualidade de vida aos cidadãos, as cidades estão cada vez mais almejando o uso combinado de várias tecnologias. Cidades Inteligentes são o resultado da junção de [Postscapes 2015]: a) sensores de baixa potência e baixo custo; b) acesso à comunicação, principalmente redes sem fio; c) engajamento da população principalmente através do uso de aplicativos para smartphones. Neste contexto, é imperativo desenvolver pesquisas, formar recursos humanos e atuar de forma integrada com a sociedade, pois a previsão é de que um enorme mercado de dezenas de bilhões de dólares deve surgir com a intensificação do processo de transformação das cidades em ambientes mais inteligentes. Neste contexto, alguns conceitos e tecnologias tem papel de destacada relevância, como Internet das Coisas (*Internet of Things* - *IoT*), Big Data, redes sociais online, computação em nuvem e tecnologias de gerenciamento flexível de redes e datacenters, como Redes Definidas por Software (*Software Defined Networking* - *SDN*) e Virtualização de Funções de Rede (*Network Function Virtualization* - *NFV*), além de computação sensível ao contexto.

A existência de sensores de múltiplas naturezas nos espaços urbanos, acoplados às onipresença dos smartphones com GPS, além de recursos computacionais disponíveis na nuvem, permitem que seja possível automatizar as cidades para melhorar a qualidade de vida dos cidadãos. O ideal já tradicional de cidades inteligentes acoplado aos novos desenvolvimentos na área de Internet das Coisas pode ser viabilizado pela computação urbana para criar sociedades mais inteligentes. Sociedades inteligentes oferecem soluções para promover a cidadania, a sustentabilidade, a integração social, a inclusão e a participação do cidadão através do uso de tecnologias inovadoras [Monroy-Hernández *et al.* 2013]. Quando os princípios das tecnologias da informação e comunicação são aplicados aos vários processos que compõem a infraestrutura urbana, o resultado é uma cidade mais inteligente [McFedries 2014b]. Cidades são inteligentes quando os investimentos em capital humano e social, assim como infraestruturas de transporte e comunicação aceleram o crescimento econômico sustentável e alta qualidade de vida

com um gerenciamento adequado dos recursos naturais através de um processo participativo de governança [Caragliu 2011].

Sociedades inteligentes em todo o mundo precisam encontrar soluções para as principais tendências que irão mudar o mundo e o modo como vivemos nas próximas décadas. Nesse contexto, uma área de importância especial para o nosso futuro é o desenvolvimento de infraestrutura urbana. Conforme Schaffers *et al.* (2011) a sociedade deveria explorar totalmente o potencial de uma infraestrutura digital inteligente que conecta pessoas, negócios e a vida urbana, suportando inovação colaborativa, consciência coletiva e criação compartilhada de serviços sustentáveis. Dentro dessa perspectiva, a União Europeia está financiando o desenvolvimento de plataformas computacionais dentro da iniciativa *Collective Awareness Platforms for Sustainability and Social Innovation*[1], que são sistemas TIC para alavancar o "efeito de rede" para criar novas formas de inovação social através da combinação de mídias sociais online abertas e dados e conhecimentos distribuídos advindos de ambientes reais .

Na sequência, a seção 4.1 prossegue detalhando conceitos de cidades inteligentes e computação urbana. A seção 4.2 apresenta as principais aplicações e desafios da computação urbana e das cidades inteligentes, enquanto que a seção 4.3 apresenta os principais conceitos e tecnologias disponíveis. Na seção 4.4. são apresentados alguns exemplos de cidades inteligentes pelo mundo com seus principais serviços e tecnologias. A seção 4.5 apresenta alguns exemplos de pesquisas em tecnologias para viabilizar cidades inteligentes e a seção 4.6 discute oportunidades e desafios de pesquisa na área. Finalmente, a seção 4.7 conclui o texto.

2.1.1. Cidades Inteligentes

O conceito de sociedades inteligentes (*smart societies*) visa criar soluções que promovam a cidadania, a sustentabilidade, à integração social, a inclusão e a participação do cidadão, através do uso de tecnologias inovadoras . Na década passada ocorreu uma transformação marcante no uso diário que as pessoas fazem da tecnologia. Antes confinado às residências e organizações, avanços em comunicações sem fio e computação móvel levaram o seu da tecnologia às ruas. Eles se tornaram parte integral da vida das pessoas, o que modifica de maneira significativa vários aspectos da sociedade, tornando-a mais inteligente e conectada, e até mesmo o envolvimento público e as ações coletivas [Monroy-Hernández *et al.* 2013].

Intimamente relacionado a sociedades inteligentes está o conceito de cidades inteligentes (*smart cities*), para o qual ainda não existe uma definição única amplamente aceita. Em geral acredita-se que cidades são inteligentes quando os investimentos em capital humano e social, assim como infraestruturas de transporte e comunicação aceleram o crescimento econômico sustentável e alta qualidade de vida com um gerenciamento adequado dos recursos naturais através de um processo participativo de governança [Caragliu 2011]. Segundo a iniciativa Smart Cities, do IEEE, uma cidade inteligente congrega tecnologia, governo e sociedade para viabilizar algumas características: uma economia inteligente, mobilidade inteligente, ambiente inteligente, pessoas inteligentes, existências inteligente, governo inteligente[2].

[1] http://ec.europa.eu/digital-agenda/en/collective-awareness-platforms

[2] http://smartcities.ieee.org

Sob a perspectiva das tecnologias da informação e da comunicação, a inteligência de um sistema pode ser caracterizada pela sua flexibilidade, adaptabilidade, memória, aprendizagem, dinâmica temporal e atuação perante a incerteza e a informação precisa. Em uma cidade inteligente, a produção, gestão e difusão do conhecimento são essenciais, bem como o gerenciamento das próprias cidades e dos dados a ela relacionados. Basicamente, se os princípios das tecnologias da informação e comunicação forem aplicados aos vários processos que compõem a infraestrutura urbana, o resultado é uma cidade inteligente [McFedries 2014b].

Entre as principais áreas de aplicação de cidades inteligentes estão incluídas a economia da inovação, a infraestrutura e serviços de utilidade da cidade e a sua própria governança [Schaffers *et al.* 2011]. A primeira área inclui os *clusters* produtivos em áreas estratégicas para a cidade; os distritos tecnológicos envolvendo empresas, universidades e até aeroportos; as estratégias para criação de novas empresas na área de tecnologia. A segunda área envolve: sistemas de transporte, mobilidade e estacionamento; ampla oferta de conectividade cabeada e sem fio, envolvendo inclusive acesso gratuito a redes WiFi; uso eficiente de energia nos seus vários aspectos, incluindo a rede elétrica inteligente (*smart grid*); monitoramento ambiental, inclusive sistemas de alerta para prevenção de catástrofes e segurança. A terceira área inclui serviços de governo eletrônico, processos de decisão participativos e sistemas de monitoramento e medição que fornecem informações online ao cidadão. Esta lista de áreas não é exaustiva, mas oferece uma visão consistente de cenários que podem ser explorados no desenvolvimento de sociedades mais inteligentes.

2.1.2. Computação Urbana

No contexto de cidades inteligentes, um conceito recente é o de computação urbana (*urban computing*), onde a cidade pode ser vista como um sistema distribuído [McFedries 2014a]. Computação urbana é o processo de aquisição, integração e análise de uma quantidade grande e heterogênea de dados gerados por diferentes fontes como sensores, dispositivos, veículos, prédios e pessoas, para tratar de questões críticas para as cidades, como poluição, consumo de energia e congestionamentos [Zheng *et al.* 2014]. Conecta o monitoramento urbano com gerenciamento e análise de dados num processo de melhoria constante da vida das pessoas, dos sistemas que operam a cidade e do meio ambiente. Computação urbana é uma área essencialmente interdisciplinar, porque necessita que a ciência da computação esteja relacionada com áreas tradicionalmente ligadas à gestão das cidades, como transportes, engenharia civil, ambiente, economia, ecologia e sociologia.

Do ponto de vista individual, o pedestre é um cursor, o cenário urbano é a interface que é clicada através de um smartphone [McFedries 2014a]. O smartphone é um componente importante em computação urbana, mas apenas como ferramenta e não como objetivo final. Através de sistemas de navegação social que envolvem o uso do onipresente GPS, é possível saber a localização de praticamente qualquer local numa cidade. Este processo é auxiliado por outras pessoas conhecidas e desconhecidas, numa rede social de ajuda mútua, como é o caso do Waze . Do ponto de vista da gestão das cidades, tornar o espaço urbano mais participativo e aumentar a qualidade de vida com recursos limitados é um fator que impulsiona para a informatização das cidades [McFedries 2014b].

Zheng *et al.* (2015) definem uma visão da computação urbana como sendo a resolução de grandes desafios (*big challenges*) de grandes cidades (*big cities*) usando grandes quantidades de dados (*big data*). Neste mesmo artigo, os autores definem um arcabouço constituído de quatro camadas: sensoriamento urbano, gerenciamento de dados urbanos, análise de dados e oferta de serviços, como mitigação de congestionamentos, economia de energia e redução da poluição. As aplicações de computação urbana podem ser divididas em diferentes categorias, como planejamento urbano, sistemas de transporte, cuidados com o ambiente, consumo de energia, aplicações sociais e segurança pública.

A computação urbana é aplicada em espaços públicos onde milhares ou milhões de pessoas convivem diariamente. A necessidade de recursos computacionais e de comunicação pode ser imensa, com padrões diferenciados de acordo com vários critérios e sujeitos a influencias sazonais, devido à própria dinâmica de mobilidade, uso e interesses das pessoas. Alguns processos dentro da computação urbana podem ocorrer off-line, como o planejamento urbano, mas outros necessitam de atenção em tempo real. Em tais ambientes, o uso compartilhado de recursos computacionais em ambientes de datacenter é essencial para tornar factível a implantação das cidades inteligentes. Estes recursos precisam ser gerenciados dinamicamente de maneira flexível para atender uma grande diversidade de aplicações, com diversas tecnologias como computação em nuvem, redes definidas por software e virtualização de funções de rede.

2.2. Aplicações e Desafios

Antes de apresentar as tecnologias frequentemente usadas em computação urbana, serão listadas sete categorias de cenários de computação urbana: planejamento urbano, sistemas de transporte, ambiente, segurança, consumo de energia, economia e aplicações sociais. Foram selecionadas algumas aplicações representativas de cada categoria, focando principalmente em seu objetivo, motivação, resultados, e os dados utilizados.

2.2.1. Planejamento Urbano

Um planejamento eficaz é de grande importância para a construção de uma cidade inteligente. Formular um planejamento urbano exige avaliação de uma ampla gama de fatores, tais como o fluxo de tráfego, mobilidade humana, pontos de interesse e estruturas da rede rodoviária. Esses fatores evolutivos complexos e rápidos transformam o planejamento urbano em uma tarefa muito desafiadora. Tradicionalmente, os planejadores urbanos dependem de trabalhos de levantamentos intensivo para informar sua tomada de decisão. Por exemplo, para entender os padrões de deslocamento urbano, uma série de trabalhos tem sido feitos com base em dados de pesquisas de viagens [Hanson e Hanson 1980; Gandia 2012; Jiang *et al.* 2012].

- Problemas subjacentes de redes de transporte: Recentemente, os amplamente disponíveis dados de mobilidade urbana gerados em espaços urbanos, na verdade, refletem os problemas subjacentes de uma cidade (dificuldade para se transportar de um ponto a outro), proporcionando aos planejadores urbanos oportunidades para melhorar planejamentos urbanos futuros. Zheng *et al.* (2011) observaram os problemas subjacentes na rede de transporte de Pequim através da análise das trajetórias de GPS gerados por 33.000 táxis durante um período de três anos. Ao comparar os resultados detectados a partir de dois anos consecutivos, a pesquisa pode até mesmo avaliar os

benefícios de um transporte recém-construído. No exemplo citado, um problema subjacente detectado em 2010 desapareceu em 2011 devido a uma linha de metrô recém-lançada. Em suma, a linha de metrô funcionou bem na resolução do problema.

- Descobrimento de regiões funcionais: O desenvolvimento de uma cidade gradualmente promove diferentes regiões funcionais, tais como áreas de educação e de negócios, que suportam diferentes necessidades da vida urbana das pessoas e servem como uma técnica de organização valiosa para enquadrar o conhecimento detalhado de uma metrópole. Estas regiões podem ser projetadas artificialmente pelos planejadores urbanos, ou naturalmente formuladas de acordo com o estilo de vida real das pessoas, mudando suas funções e territórios com o desenvolvimento da cidade. A compreensão das regiões funcionais em uma cidade pode calibrar o planejamento urbano e facilitar outras aplicações, como a escolha de um local para um negócio. Yuan e Zheng *et al.* (2012) propuseram um *framework* (intitulado DRoF) que descobre regiões de diferentes funções em uma cidade. Como resultado, uma região é representada por uma distribuição de funções e cada uma delas ainda é indicada por uma distribuição de padrões de mobilidade, mostrando pessoas deixando áreas residenciais de manhã e voltando à noite, por exemplo.

- Detectando os limites da cidade: As fronteiras regionais definidas pelos governos podem não respeitar as formas naturais que as pessoas interagem através do espaço. A descoberta das fronteiras reais de regiões de acordo com a interação entre as pessoas pode fornecer ferramentas de apoio à decisão para os políticos, sugerindo uma fronteira administrativa ideal para uma cidade. A descoberta destes limites também ajuda o governo entender a evolução do território de uma cidade. A ideia geral desta categoria de pesquisa é primeiro a construir uma rede entre locais com base na interação humana, por exemplo, trilhas de GPS ou registros de chamadas de telefone e, em seguida, particionar a rede usando algum método de descoberta de comunidade, que encontra grupos com maior interação local. Ratti *et al.* (2010) propôs uma abordagem refinada para delimitação regional, através da análise da rede humana inferida de um grande banco de dados de telecomunicações na Grã-Bretanha. O algoritmo resultou em regiões geograficamente coesas que correspondem com as regiões administrativas e revelou inesperadas estruturas espaciais que tinham hipótese apenas na literatura.

2.2.2. Sistemas de Transporte

- Melhorar as experiências de condução: Encontrar percursos de condução rápida economiza tanto o tempo do motorista e consumo de energia quanto o congestionamento do tráfego, reduzindo a quantidade de emissão de gases poluentes [Hunter *et al.* 2009; Kanoulas *et al.* 2006]. Estudos intensivos têm sido feitos para aprender padrões históricos de tráfego [Bejan *et al.* 2010; Herrera *et al.* 2010], estimar fluxos de tráfego em tempo real [Herring *et al.* 2010], e prever futuras condições de tráfego [Castro-Neto *et al.* 2009] em segmentos de rodovias individuais com dados oscilantes [Pfoser 2008a; 2008b], tais como trajetos de GPS dos veículos, sinais de WiFi e de telefonia celular. No entanto, o trabalho de modelagem do padrão de tráfego em toda a cidade ainda é raro. VTrack [Thiagarajan *et al.* 2009] é um sistema para a estimativa de tempo de viagem baseado no sinal WiFi, mensurando e localizando os tempos de atraso. O sistema utiliza um modelo oculto de Markov (HMM) - esquema de combinação de mapas que interpola dados esparsos para identificar os segmentos de estrada mais prováveis dirigidos pelo usuário. Os experimentos mostram que VTrack

pode tolerar ruídos e interrupções significativas nestas estimativas de localização, e ainda identificar com sucesso segmentos propensos a atrasos.

- Melhorar os serviços de táxi: Taxi é um modo de comutação importante entre transportes públicos e privados. Em grandes cidades como Nova York e Pequim, as pessoas costumam esperar um tempo não-trivial antes de encontrar um táxi vago, enquanto os motoristas de táxi estão ansiosos para encontrar passageiros. Assim, conectar passageiros com táxis vagos é de grande importância para a economia de tempo de espera das pessoas, para aumentar o lucro dos motoristas de táxi, e para reduzir o tráfego desnecessário e consumo de energia. Há três categorias para abordar esta questão:

a) Sistemas de envio de táxi: Este tipo de sistema [Lee et al 2004] aceita o pedido de reserva de um usuário e envia um táxi para ele. A maioria dos sistemas solicitam reservas de táxi com antecedência, reduzindo a flexibilidade do serviço. Alguns sistemas procuram em tempo real táxis livres em torno de um usuário, com base no princípio do vizinho mais próximo em distância e tempo. O principal desafio é considerar a incerteza da circulação dos táxis [Phithakkitnukoon *et al.* 2010; Yamamoto *et al.* 2010].

b) Sistemas de recomendação para taxistas: Esta categoria de sistemas aborda o problema da perspectiva da recomendação. Ge *et al.* [2010] desenvolveu um sistema que tem a capacidade de recomendar uma sequência de pontos de recolha para os motoristas de táxi ou uma sequência de posições de estacionamento potenciais. O objetivo do sistema é para maximizar a probabilidade de sucesso do negócio e reduzir o consumo de energia. O principal desafio desta categoria é para lidar com o problema de dispersão de dados.

c) Serviços de compartilhamento de táxi: Esta categoria é de grande importância para a economia no consumo de energia e para aliviar o congestionamento do tráfego. T-Share [Ma *et al.* 2013] é um sistema dinâmico de compartilhamento de táxi em larga escala que aceita pedidos de carona em tempo real dos passageiros. O sistema cria um cenário que produz benefícios sociais e ambientais significativos. De acordo com uma simulação baseada na trajetória de 30.000 táxis em Pequim, o sistema seria capaz de economizar 120 milhões de litros de gasolina por ano e reduzir 246 milhões de kg de CO2 emitidos. Além disso, os passageiros economizam 7% na tarifa de táxi e têm 300% mais chances de conseguir um táxi, enquanto a renda dos taxistas aumenta 10% [Ma *et al.* 2013]. A dificuldade de alcançar tal sistema de compartilhamento de táxi reside em dois aspectos. Um deles é o problema de modelar o tempo, capacidade e restrições monetárias para viagens de táxi. A outra é a pesada carga computacional causada pela dinâmica de passageiros e de táxis, o que pede algoritmos de busca eficientes. Claro que, na prática, ainda existem outros problemas não técnicos que tem de ser resolvidos, por exemplo, criar um medidor de qualidade de passageiro e taxista, bem como alguns problemas de segurança

- Melhorar os sistemas de transporte público: Em 2050 espera-se que 70% da população mundial estará vivendo em cidades. Enfrentaremos um mundo cada vez mais urbanizada e poluído. A construção de sistemas mais eficazes de transporte público, como alternativa aos veículos privados, tornou-se prioridade, para proporcionar uma boa qualidade de vida, um ambiente mais limpo, e manter-se economicamente atraente

para os potenciais investidores e funcionários. Sistemas de transporte público de massa são considerados elementos fundamentais para melhor a mobilidade. Algumas das mais recentes aplicações de computação urbana de transportes públicos são: ônibus, metrô, e compartilhamento de bicicletas.

1) Serviços de ônibus: A fim de atrair mais passageiros, os serviços de ônibus precisam ser mais frequentes e também mais confiáveis. Watkins *et al.* [2011] realizou um estudo sobre o impacto do fornecimento de informações de chegada de ônibus em tempo real diretamente nos celulares dos passageiros, para reduzir o tempo de espera no ponto de ônibus. Os resultados mostraram usuários mais satisfeitos com o uso dos serviços de ônibus.

2) Serviços de metrô: sistemas automatizados de cobrança foram introduzidos e são amplamente utilizados em muitas cidades metropolitanas em todo o mundo. Além de simplificar o acesso à rede de metrô da cidade, esses cartões inteligentes criam um registro digital cada vez que uma viagem é feita. Minerar os dados de viagens pode revelar muito sobre os viajantes: as suas preferências implícitas, tempos de viagem e hábitos. Lathia *et. al* [2010] minerou dados do sistema de metrô de Londres com o objetivo de construir uma rota de viagem mais precisa para os planejadores.

3) Sistemas de compartilhamento de bicicletas: Como a população mundial crescendo e a proporção de pessoas que vivem em cidades aumentando, projetar, manter e promover o desenvolvimento urbano e sustentável de modos de mobilidade está se tornando de suma importância. Esquemas de bicicletas compartilhadas [Shaheen *et. al* 2010] são um exemplo: a sua proliferação em todas as metrópoles do mundo claramente reflete a crença de que proporcionando fácil acesso a modos saudáveis, e rápidos, de transporte levará as cidades para longe dos problemas de congestionamento e da poluição. Registros detalhados estão frequentemente disponíveis sobre "de onde / quando uma bicicleta foi tirada", para "onde / quando uma bicicleta foi devolvida", permitindo assim que os investigadores analisem estes rastros digitais para ajudar os usuários finais

2.2.3. Ambiente

Sem um planejamento eficaz e adaptável, um rápido progresso da urbanização irá tornar-se uma ameaça potencial ao meio ambiente das cidades. Recentemente, temos assistido a uma tendência crescente de contaminações em diferentes aspectos do ambiente, tais como a qualidade do ar, ruído e lixo, em todo o mundo. Proteger o meio ambiente ao mesmo tempo modernizar a vida das pessoas é de suma importância na computação urbana.

- Qualidade do Ar: Informações sobre a qualidade do ar urbano, por exemplo, a concentração de PM2,5 (aerossóis), é de grande importância para a proteção da saúde humana e para controlar a poluição do ar. Muitas cidades estão monitorando PM2,5 através da construção de estações de medição da qualidade do ar no solo. No entanto, existem apenas um número limitado de estações de medição da qualidade do ar em uma cidade, devido ao alto custo de construção e manutenção. A qualidade do ar varia de acordo com os locais de forma não linear e depende de vários fatores, como a meteorologia, o volume de tráfego e uso da terra. Como resultado, não há como saber a qualidade do ar de um local sem uma estação de medição. Os avanços em tecnologias de comunicação e sensores móveis têm proliferado os aplicativos baseados em

crowdsourcing, que decompõem um problema complexo em pequenas tarefas e distribui essas pequenas tarefas para uma rede de usuários. Os retornos de usuários individuais formularão o conhecimento coletivo que pode resolver o problema complexo. Copenhagen Wheels é um projeto que instala sensores ambientais na roda de bicicletas para monitorar dados ambientais como temperatura, umidade e concentração de CO2. O trabalho humano para andar de bicicleta é transferido para poder apoiar o funcionamento dos sensores. Além disso, a roda pode se comunicar com o telefone móvel de um usuário, através do qual a informação recolhida é enviada para um sistema de *backend*.

- Poluição sonora: As funções compostas de uma cidade e suas configurações complexas que incorporam diferentes infraestruturas e milhões de pessoas, têm gerado uma grande quantidade de ruído ambiental. Como resultado, uma grande quantidade de pessoas, em todo o mundo, está exposta a altos níveis de poluição sonora, que podem causar graves doenças que variam de deficiência auditiva, influências negativas na produtividade humana e comportamento social [Rana *et. al* 2010]. Como uma estratégia de redução, uma série de países, como os EUA, o Reino Unido, e na Alemanha, começaram a monitorar a poluição sonora. Eles geralmente usam um mapa de ruído (a representação visual do nível de ruído de uma área) para avaliar os níveis de poluição sonora. O mapa de ruído é calculado usando simulações com base dados como de fluxo de tráfego, estradas, transportes ferroviários e veículos. Uma vez que a obtenção dos referidos dados de entrada é muito cara, estes mapas podem ser atualizados apenas após um longo período de tempo. Silvia *et al.* [2008] avalia a poluição sonora ambiental em áreas urbanas usando redes de sensores sem fio. Para implantar e manter uma rede de sensores em toda a cidade, especialmente em grandes cidades como Nova York, no entanto, é muito caro, em termos de dinheiro e de recursos humanos. Outra solução é tirar proveito de crowdsourcing, onde as pessoas recolhem e compartilham a sua informação ambiental utilizando um dispositivo móvel, por exemplo, um telefone móvel. Por exemplo, NoiseTube [Nicolas *et al.* 2009] apresenta uma abordagem que aproveita as medições de ruído compartilhadas por usuários de telefones celulares para pintar um mapa de ruído em uma cidade. Baseado em NoiseTube, D'Hondt e Stevens [2011] realizaram um experimento científico mapeando uma área de 1 km^2, na cidade de Antuérpia. O principal objetivo do trabalho foi investigar a qualidade do mapa de ruído obtido por sensoriamento participativo, em comparação com mapas oficiais baseados em simulação de ruído.

2.2.4. Segurança Pública

Grandes eventos, pandemias, acidentes graves, desastres ambientais e ataques terroristas trazem desafios adicionais para a segurança pública. A grande quantidade de diferentes tipos de dados urbanos nos permite aprender com dados históricos e prever situações de emergência.

- Detecção de anomalias no tráfego: As anomalias de tráfego em áreas urbanas podem ser causadas por acidentes, controles, protestos, esportes, celebrações, desastres, entre outros eventos. A detecção destas anomalias pode ajudar a dispersar o congestionamento, prevendo esses eventos e facilitando o deslocamento das pessoas. De acordo com [Chandola, 2009; Hodge, 2004] podemos dividir as metodologias para tratar essas anomalias em categorias: 1) Métodos baseados em distância; e 2) Métodos baseados em estatística;

- Detecção de desastre e evacuação: O acidente nuclear em Fukushima causou um grande movimento de evacuação da população. O entendimento e a prática deste tipo de situação é de extrema importância para o planejamento efetivo de socorro humanitário, administração das consequências do desastre e reconstrução da cidade a longo prazo.

2.2.5. Consumo de Energia

O rápido avanço da urbanização está fazendo com que o consumo de energia aumente, contribuindo para o desenvolvimento de novas tecnologias que possam reduzir o consumo energético e melhorem a infraestrutura disponível.

- Consumo de Gasolina: Zhang *et al.* [2013] propuseram um procedimento para a detecção em tempo real do comportamento de reabastecimento e consumo de gasolina em toda a cidade. O método utiliza uma abordagem de "humano como um sensor", analisando e fazendo inferências a partir de trajetórias de GPS passivamente coletadas por táxis. O resultado é uma estimativa global do tempo gasto e uso de combustível em cada posto de gasolina em cada período de tempo.

- Consumo de Eletricidade: A integração eficiente do uso energia a partir de fontes renováveis é a chave para a sustentabilidade do fornecimento de eletricidade. Para otimizar o uso de energia residencial, mecanismos de resposta de demanda inteligente são necessário para mudar o uso de energia para períodos de baixa demanda, ou para períodos de alta disponibilidade de energia renovável. Algoritmos inteligentes permitem que dispositivos atendam às políticas de usuários individuais e ainda fiquem dentro dos limites de uso de energia da comunidade. Em Dusparic *et. al* [2013], cada veículo elétrico dentro de uma comunidade é controlado por um agente de reforço de aprendizagem, ainda apoiada por um algoritmo de previsão de carga a curto prazo.

2.2.6. Economia

A dinâmica de uma cidade, como por exemplo a mobilidade urbana e o número de regiões funcionais, pode indicar o status da economia da cidade. Por exemplo, o número de cinemas em Beijing cresceu entre 2008 e 2012. Isso significa que cada vez mais pessoas querem ir ao cinema, mostrando que os negócios neste segmento estão prosperando. Da mesma maneira, uma quantidade crescente de pessoas se deslocando no início e fim de horários comerciais pode indicar uma diminuição de pessoas desempregadas.

As informações de mobilidade, combinadas com as informações de regiões funcionais, podem ajudar a estabelecer um ponto de fixação para um negócio, por exemplo. Karamoshuk *et. al* [2013] estudou o problema da otimização de fixação de uma loja de varejo baseado em informações de redes sociais, mais especificamente os dados de mobilidade coletados do Foursquare. Como resultado, entre outras características, a presença de facilitadores para os usuários (como estações de trem, aeroporto, etc.), assim como a presença de outras lojas do mesmo tipo (como restaurantes, cafeterias, etc.) ajudam a criar uma região comercial forte, aumentando a popularidade de um local e aumentando as chances de um novo comércio prosperar nesta região.

2.2.7. Aplicações Sociais

Embora existam muitos tipos de serviços de redes sociais na Internet, vamos nos concentrar na introdução de redes sociais baseadas em localização (LBSN), que são formalmente definidas da seguinte maneira em [Zheng 2011 e 2012]:

A rede social baseada em localização (LBSN) não significa apenas a adição de um local para uma rede social existente, mas também consiste em uma nova estrutura social composta por indivíduos ligados pela interdependência derivada de suas localizações no mundo físico, bem como o seu conteúdo de mídia marcados por uma localização, como fotos, vídeos e textos. Aqui, a localização física consiste na localização instantânea de um indivíduo associada a uma data. Além disso, a interdependência se define não apenas por duas pessoas estarem no mesmo local físico ou terem históricos semelhantes de localização, mas também o conhecimento, por exemplo, de interesses comuns, comportamentos e atividades, inferidas a partir da localização de um indivíduo. Exemplos de LBSNs incluem o Foursquare e um protótipo de pesquisa chamado GeoLife [Zheng *et al.* 2010]. Com LBSNs, podemos entender usuários e locais, respectivamente, e explorar a relação entre eles.

1) Uso de Estimativa de Semelhança: o histórico de localização de um indivíduo no mundo real implica, em certa medida, seus interesses e comportamentos. Assim, pessoas que compartilham histórias com localização semelhantes são susceptíveis de ter interesses e comportamentos comuns. A semelhança entre os usuários inferida a partir de suas histórias de localização pode permitir recomendações de amigos [Li et al 2008], que conectam os usuários com interesses semelhantes, mesmo quando eles podem não ter conhecido um ao outro anteriormente. Também há a descoberta de comunidade [Hung *et al.* 2009] que identifica um grupo de pessoas que compartilham interesses comuns. Para melhor estimar a similaridade entre os usuários, mais informações, tais como as sequências que visitam entre locais, a granularidade geoespacial de um local, e a popularidade de um local, são considerados em [Zheng *et al.* 2010].

2) Encontrar especialistas locais em uma região [Zheng *et al.* 2009]: Pessoas da região são capazes de identificar os especialistas locais por terem mais conhecimento sobre a região do que outros. Por exemplo, os especialistas locais são mais propensos a saber sobre os restaurantes de alta qualidade do que alguns turistas.

3) Recomendações de Locais: Encontrar os locais mais interessantes em uma cidade é uma tarefa que, em geral, um turista quer cumprir ao viajar para uma cidade desconhecida [Zheng *et al.*2009]. No entanto, o nível de interesse de um local não depende do número de pessoas que visitaram o local, mas também do conhecimento de viagens dessas pessoas. Por exemplo, o local mais frequentemente visitado em uma cidade poderia ser sua estação ferroviária ou aeroporto que pode não ser uma localização interessante para recomendação.

2.3. Conceitos e Tecnologias

2.3.1. Internet das Coisas

A Internet das Coisas [Atzori *et al.* 2010] pode ser vista como o próximo passo da Internet, em que objetos inteligentes se conectam em uma infraestrutura de rede global.

Embora não possua uma definição única, a IoT permite que as pessoas e coisas se conectem a qualquer hora, em qualquer lugar, utilizando algum tipo de rede e protocolos padrões de comunicação para utilizar algum serviço [Razzaque *et al.* 2016]. As "coisas" que compõe a IoT incluem uma grande variedade de dispositivos, como computadores, smartphones e tablets; sensores e atuadores; e outros objetos do cotidiano como eletrodomésticos e relógios e até veículos, entre outros. Alguns dos componentes essenciais da IoT são as redes de sensores sem fio, sistemas RFID (*Radio Frequency Identification*) e comunicações M2M (máquina-a-máquina) [Atzori et al 2010]. Os sistemas RFID são compostos de leitores e etiquetas RFID. Estas etiquetas são colocadas em objetos e possuem um identificador único. As redes de sensores sem fio são formadas por vários nós sensores que fazem o sensoriamento de uma região e enviam estas informações para um nó coletor, geralmente com conexão à internet. Existe vasta literatura disponível sobre estas tecnologias, mas a integração de todas elas em uma rede única traz vários desafios para a Internet das Coisas.

Uma das características mais marcantes da IoT é a heterogeneidade dos dispositivos. Os objetos possuem diferentes capacidades de processamento, armazenamento e comunicação. Enquanto alguns são bastante limitados (ex.: sensores), outros não têm estas mesmas limitações (ex.: veículos). Para os dispositivos embarcados com limitações computacionais e de energia, os protocolos e aplicações devem ser otimizados para não esgotar os recursos disponíveis. Entre as tecnologias de comunicação da IoT para estes dispositivos, estão o *Bluetooth* e *Bluetooth Low Energy* (BLE), IEEE 802.15.4, PLC (*Power Line Communications*), RFID e NFC (*Near Field Communications*). Para os dispositivos sem limitações, podem ser utilizados Ethernet, WiFi, fibra ótica e tecnologias celulares como LTE (*Long Term Evolution*).

As comunicações entre os dispositivos frequentemente ocorrem de maneira espontânea, sem intervenção dos usuários. Como o número de dispositivos conectados que interagem pode chegar a centenas ou até milhares em único ambiente ou alguns bilhões em toda a IoT, várias questões de escalabilidade são importantes. Por exemplo, o endereçamento dos dispositivos, usando protocolos como IPv6 e 6LoWPAN (*IPv6 over Low Power Wireless Personal Area Networks*). Os eventos gerados pela interação dos objetos produz uma enorme quantidade de dados, e precisa ser tratada adequadamente. Em geral, a arquitetura de rede da IoT não possui infraestrutura e é altamente dinâmica, pois os nós entram e saem da rede com frequência. Ainda, os nós podem ser desconectados por falta de energia ou por enlaces sem fio com baixa qualidade de sinal. Logo, a rede precisa ter capacidade de auto-organização [Razzaque et al 2016].

As aplicações em IoT precisam ser cientes de localização e de contexto, inteligentes e distribuídas [Perera *et al.* 2014]. Os dados gerados por sensores e dispositivos RFID, por exemplo, precisam ser analisados e interpretados dependendo do contexto em que estão inseridos, de preferência sem intervenção humana. A ciência de localização dos objetos e sensores é essencial para as aplicações cientes de contexto. Assim, tem-se a visão de dispositivos inteligentes que precisam atuar de maneira independente e que se comunicam e fornecem dados para sistemas inteligentes. As aplicações usam arquiteturas e componentes orientadas a serviço, como serviços web e protocolos como CoAP (*Constrained Application Protocol*), MQTT (*Message Queue Telemetry Transport*) e HTTP (*Hyper Text Transfer Protocol*).

A IoT fornece diferentes serviços para diversos ambientes, entre eles o ambiente urbano [Zanella *et al.* 2014]. Uma IoT urbana pode colaborar em diversas áreas, como sistemas de transporte e estacionamento, iluminação pública, coleta de lixo, distribuição de água e energia elétrica, entre outros. A estrutura de rede e comunicação da IoT pode prover aos cidadão o acesso aos mais diversos serviços públicos, ajudando na melhoria da qualidade dos mesmos.

2.3.2. Computação em Nuvem e Computação em Névoa

A computação em nuvem proporciona ao usuário serviços de computação como utilidade pública [Armbrust *et al.* 2010]. Para o usuário, não há necessidade de tomar conhecimento onde os dados são armazenados e processados, assim como com questões de provisionamento de recursos para atender a demanda. O usuário tem uma necessidade, usa os recursos computacionais do seu provedor de nuvem computacional e paga exatamente pela quantidade de recursos utilizados. Os modelos de entrega de serviços podem ser classificados em três categorias: software como serviço (*SaaS*), plataforma como serviço (*PaaS*), e infraestrutura como serviço (*IaaS*). Virtualização de recursos computacionais é uma tecnologia que desempenha papel importante em computação em nuvem. Um serviço como o Amazon AWS representa uma nuvem pública, mas também existem várias alternativas para usar serviços de nuvem, principalmente IaaS em ambientes privados, formando nuvens privadas. Estas últimas podem ainda ser usadas em conjunto com nuvens públicas para aumentar sua capacidade de processamento, formando assim nuvens híbridas.

O fornecimento destes serviços de computação em nuvem com qualidade, segurança e custo competitivo por *datacenters* especializados representa um grande desafio, tanto tecnológico como de mercado. Entre os inúmeros desafios neste cenário, a gerência de recursos computacionais neste ambiente exige cada vez mais soluções inovadoras e tecnologias experimentais, fornecendo um fluxo constante de oportunidades de pesquisa [Batista *et al.* 2011]. Essa tecnologia é apoiada sobre dois grandes pilares: computadores e redes. Entende-se por computadores os servidores, responsáveis pelo processamento dos dados, e os dispositivos de armazenamento (*storage*), responsáveis pelo armazenamento destes dados. Essa tecnologia, considerada um commodity, teve significativo avanço na integração de dispositivos digitais e os softwares responsáveis pelo uso desse hardware, com destaque para a tecnologia de virtualização, peça fundamental para prover recursos de computação em nuvem, além do surgimento dos softwares controladores de nuvem computacional, como o OpenStack[3], OpenNebula[4], Ganeti[5], Nimbus[6], entre outros. Entre esses projetos, o OpenStack [Sefraoui *et al.* 2012] tem ganhado grande notoriedade nos últimos anos. Além de parceiros como IBM, Intel, AT&T, HP, entre outros, mostrou-se uma solução poderosa e robusta para o gerenciamento e orquestração, tanto de nuvens públicas como privadas. Além dos recursos básicos fornecidos em um ambiente de computação em nuvem, como o fornecimento de máquinas virtuais e espaço de armazenamento, o OpenStack, atualmente, conta com recursos sofisticados, desde o gerenciamento de

[3] http://openstack.org
[4] http://www.opennebula.org
[5] http://code.google.com/p/ganeti
[6] http://www.nimbusproject.org

objetos complexos e Big data até o controle dos recursos disponibilizados, ou não, ao cliente final da nuvem computacional. Outra característica é o amplo suporte a tecnologias de virtualização, comunicação e hardware especializado disponíveis no mercado.

Nuvens públicas são tipicamente implementadas em datacenters localizados em alguns pontos do planeta e acessadas universalmente através da Internet. Essa solução oferece os serviços convencionais aos usuários mas posiciona os servidores físicos onde os serviços são implementados em locais distantes dos clientes que os utilizam. Por esse motivo, existe uma tendência de levar serviços de processamento e armazenamento mais próximos do usuários, para as bordas da rede [Bonomi *et. al* 2012], para possibilitar a criação de novas aplicações e serviços, conhecida como Computação em Névoa (*Fog Computing*). Algumas características de Fog são: a) baixa latência; b) distribuição geográfica; c) mobilidade; d) grande número de nós conectados; e) acesso através de redes sem fio; f) uso de aplicações de streaming e tempo real; g) heterogeneidade. Além de beneficiar usuários com essas características, Fog está sendo considerada para possibilitar o desenvolvimento de cenários que envolvem a oferta de novos serviços e aplicações baseados na Internet das Coisas (IoT), inclusive Cidades Inteligentes.

Cidades Inteligentes requerem uma variedade de novos serviços em ambientes urbanos, baseados em dados oriundos de fontes diferentes. Isto coloca demandas altas para a alocação dinâmica da infraestrutura de computação e comunicação que atualmente é baseada em datacenters de nuvem. O gerenciamento de mecanismos de configuração dinâmica para elasticidade, SDN e NFV é uma característica chave para tornar possível a existência desses cenários [Kamienski 2015].

2.3.3. SDN e NFV: Softwarização de Redes

Atualmente existe uma tendência de que cada vez mais as infraestruturas de rede e de telecomunicações sejam totalmente definidas por software, que tem sido chamada de softwarização de redes (*network softwarization*[7]). As duas principais tecnologias indutoras dessa nova tendência são SDN (Software-Defined Networking) e NFV (Network Functions Virtualization), que são conceitos complementares [ONF 2014, ONF 2015]. SDN flexibiliza as redes, tornando possível alterações dinâmicas que atualmente precisam ser realizadas manualmente. Por outro lado, NFV flexibiliza as funções de rede (VNF) necessárias para criar os serviços de rede. NFV não necessariamente precisa ser implementada em um ambiente de datacenter com SDN e SDN por sua vez pode ter usos mais abrangentes. No entanto, ambas podem ser vistas como fazendo parte do grande cenário de redes em um datacenter e podem obter uma elevada sinergia quando acopladas.

O crescimento na demanda dos serviços ofertados pelos *datacenters*, pressionada principalmente pela ampla adoção da computação em nuvem, colocam em xeque as arquiteturas tradicionais de rede. Limites como número de VLANs, ACLs, políticas de QoS, antes inalcançáveis tornam-se insuficientes diante de infraestruturas com milhares de nós de rede. Além disso, a convivência com diversos fornecedores, cada qual com seus protocolos e tecnologias, tornam o gerenciamento da rede de *datacenter* um desafio cada vez maior.

[7] http://sites.ieee.org/netsoft/about/netsoft-2016/

Em 2008, um grupo de pesquisadores de Stanford movidos pela necessidade de um ambiente realista de experimentação para novas ideias e protocolos, bem como tornar a rede programável, apresentaram o protocolo OpenFlow [McKeown *et. al* 2008], consolidando o conceito de SDN. A tecnologia SDN propõe a separação dos planos de dados, responsável pelo encaminhamento dos pacotes, do plano de controle, responsável por definir as regras de encaminhamento dos pacotes. Tradicionalmente, estes planos são encapsulados nos dispositivos de rede. A proposta da tecnologia SDN transfere a manipulação das regras de encaminhamento para a figura do controlador, que por sua vez disponibiliza uma API para que as aplicações possam interagir com a rede.

Entre as vantagens preconizadas pela tecnologia SDN em [Open Networking Foundation 2012], destacam-se:

- Gerenciamento centralizado de dispositivos de diferentes fabricantes;
- Uso da API para otimizar e abstrair o controle da rede;
- Acesso rápido a novas tecnologias desacopladas do hardware;
- Maior nível de granularidade para definição de políticas de encaminhamento de pacotes;
- Aprimoramento da segurança da rede.

Na Figura 2.1 é apresentada a arquitetura da tecnologia SDN, definida pelo Open *Network Foundation* (ONF) [Open Networking Foundation 2012], responsável pela normatização e padronização da tecnologia SDN. Na figura, observa-se o plano de dados, responsável pelo encaminhamento dos dados entre os diversos pontos de rede – este é basicamente formado pelos *switches* com tecnologia SDN. No centro tem-se o controlador, responsável por definir as regras de encaminhamento e comunicar-se com os *switches* atualizando suas tabelas de fluxo e inspecionando o tráfego se necessário. É responsabilidade também do controlador fornecer a API para os aplicativos que desejam utilizar este recursos fornecidos pela tecnologia.

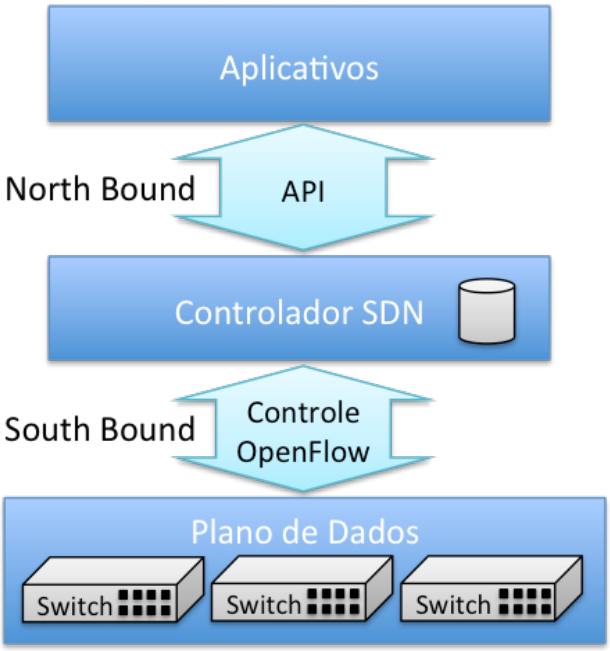


Figura 2.1. Arquitetura da Tecnologia SDN

Sob premissas similares, em 2012 o *European Telecommunications Standards Institute* (ETSI) propôs, em 2012, o conceito de NFV [ETSI *et al.* 2012]. O trajeto percorrido pela informação nas redes atuais envolve outros elementos além dos enlaces. Esses elementos conhecidos como middleboxes, sob o ponto de vista da virtualização de redes, são denominados funções de rede. Essas funções de rede são, via de regra, implementadas por equipamentos dedicados, com hardware e software proprietários, conhecidos como *appliances*.

Além do alto custo de aquisição desses equipamentos, as tecnologias utilizadas impedem a evolução e/ou modificação dos mesmos para implementar novas ideias e tecnologias experimentais. Do ponto de vista do gerenciamento, por vezes, é realizado apenas no local via console, além de exigir modificações nas conexões físicas em certas circunstâncias. Finalmente, o dimensionamento dos mesmos é realizado considerando a demanda máxima, gerando desperdícios na aquisição dos equipamentos e no consumo de energia. Quando essa demanda máxima é superada, novos equipamentos devem ser adquiridos, configurados e instalados no local, gerando um tempo de solução para o problema da demanda que pode compreender algo entre horas e dias.

A Figura 2.2 mostra o conceito básico que apoia a tecnologia NFV, ou seja, a substituição de equipamentos dedicados por versões virtuais suportadas por servidores e equipamentos de rede commodities utilizando virtualização. Nesse novo paradigma, a Função de Rede Virtualizada, ou *Virtualizated Network Function* (VNF), é implementada em máquinas virtuais personalizadas visando obter o máximo de desempenho para a função à qual foi designada. Quando uma necessidade de infraestrutura precisa ser satisfeita, uma nova instância dessa máquina virtual é criada – quando esta não for mais necessária, a mesma é destruída, liberando os recursos computacionais do equipamento hospedeiro.

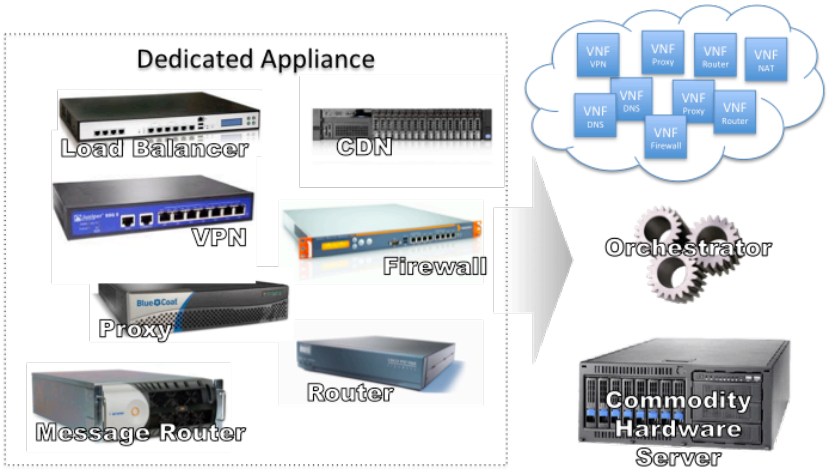


Figura 2.2. Conceito de NFV

Entre os benefícios dessa nova tecnologia apontados em ETSI (2012), destacam-se:

- Redução de custo de equipamentos e de consumo de energia;
- Redução no tempo necessário para implementar novas abordagens/tecnologias de rede desacoplando o hardware no processo de inovação;

- Uso de plataformas unificadas para fornecer diversos tipos de serviços, compartilhando a mesma base em diferentes implementações de serviços de rede;
- Os recursos podem ser direcionados para um conjunto de clientes ou com base em posição geográfica, tornando a escalabilidade dos serviços mais eficiente;
- A grande variedade de ecossistemas incentivam a abertura dos padrões, permitindo uma maior concorrência, encorajando a iniciativa de pequenas empresas na concorrência neste mercado.

As tecnologias de computação em nuvem, SDN e NFV permitem a implementação dos conceitos de *Infrastructure as a Service* (IaaS) e *Network as a Service* (NaaS) de forma ampla, através da abstração da rede oferecida pela tecnologia SDN, pela abstração das funções oferecida pela tecnologia NFV e pela abstração da computação oferecida pela tecnologia de Computação em Nuvem. A Figura 2.3 mostra o relacionamento entre estas três tecnologias que, apesar de independentes, beneficiam-se mutuamente.

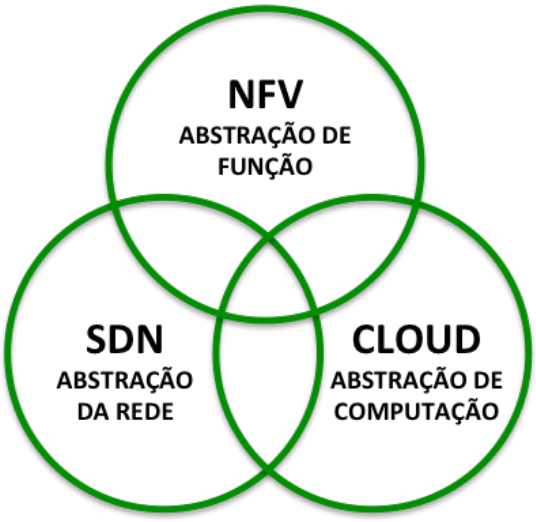


Figura 2.3. Relação entre NFV, SDN e Computação em Nuvem. Adaptado de Mijumbi *et al.* (2015)

A utilização destas tecnologias em um ambiente de *datacenter* permite a seus administradores a visão global de sua infraestrutura, centralizando o controle e simplificando a modificação de sua organização lógica, melhorando consequentemente a qualidade do serviço oferecido e o custo, redução de custo operacional e impacto ambiental. Estas características são compatíveis com o ambiente altamente dinâmico e de demanda variável típico de cidades inteligentes.

2.3.4. Mobilidade e Redes sem Fio

Parte significativa das tecnologias envolvidas, tanto na coleta de dados como na atuação remota, envolve algum tipo de tecnologia de comunicação sem fio. Seja pela dificuldade em fornecer infraestrutura cabeada em determinados pontos da cidade ou pela própria característica móvel do elemento monitorado, as redes sem fio são fundamentais para o cenário de cidades inteligentes. Exemplos deste tipo de aplicação

são pequenas estações meteorológicas alimentadas por energia solar em pontos periféricos da área urbana transmitindo dados por algum tipo de tecnologia sem fio.

As redes sem fio tradicionais são infraestruturadas, como as redes celulares e Wi-Fi, em que uma estação base ou ponto de acesso fornecem o acesso aos clientes a uma infraestrutura de rede mais ampla, como a internet. As redes ad hoc móveis sem fio, ou MANETs (*Mobile Ad hoc Networks*), são redes formadas para um propósito específico com múltiplos nós com liberdade de movimento e não tem a necessidade de um ponto de acesso [Tonguz *et al.* 2006]. No contexto das soluções de conectividade utilizadas em cidades inteligentes, as redes infraestruturadas e ad hoc apresentam uma enorme variedade de protocolos e tecnologias de transmissão e recepção de dados, técnicas de modulação, utilizam diferentes faixas do espectro de radiofrequência, além dos problemas de coexistência pelo compartilhamento do meio físico.

O IEEE mantém vários grupos de trabalho com o objetivo de fomentar e regulamentar as tecnologias de comunicação sem fio, entre as quais se destacam o 802.16 (WiMAX) direcionado a redes metropolitanas e o 802.15 voltado a redes pessoais sem fio (WPANs), incluindo os padrões 802.15.1 (Bluetooth) e 802.15.4 (ZigBee). A tecnologia de redes sem fio mais usada é sem dúvida a 802.11, popularmente conhecida com WiFi. Esta tecnologia está presente em prédios públicos e particulares, praças públicas, nos smartphones, câmeras de segurança e muitos outros dispositivos relacionados aos conceitos de cidades inteligentes. O protocolo 802.11, bem como suas extensões, utilizam a ideia proposta pelos protocolos padrão 802 de dividir a camada de enlace de dados (*Data Link Layer*) em duas subcamadas: *Logical Link Control* (LLC) e *Media Access Control* (MAC). O protocolo 802.11 e suas variantes atuam exatamente na subcamada MAC, tornando a operação da rede sem fio transparente para as camadas superiores da pilha de protocolos do modelo OSI. Esta característica promoveu a ampla adoção por muitos fabricantes do protocolo 802.11 e a sua consequente evolução em termos de robustez, taxa de transmissão e redução de custo.

Outro meio de comunicação de dados nas cidades inteligentes é o uso das redes de dados 4G e 5G associadas à telefonia celular. A evolução destas redes que além de seu uso convencional vem sendo adotada para monitoramento e atuação remota é de suma importância para as cidades inteligentes. A taxa de crescimento no tráfego de dados nestas redes é estimada, entre 2012 e 2016, em 78% ao ano, e a indústria espera para os próximos seis anos um crescimento de até 1000 vezes neste tráfego [Chen e Zhao 2014]. A grande vantagem do uso desta tecnologia em relação ao 802.11 é a possibilidade de grandes distâncias entre o dispositivo e a estação base e a mobilidade deste dispositivos em uma área urbana de grandes dimensões pois os mesmos estão associados à organização da rede de dados da operadora de telefonia móvel e não de uma rede local. Recentemente um novo tipo de conectividade sem fio tem sido proposto, com características de baixa taxa de dados e longo alcance, chamadas de LPWANs (*Low Power Wide Area Networks*). Estas novas tecnologias, como LoRa, são bastante promissoras para aplicações de cidades inteligentes [Centenaro *et al.* 2015].

2.3.5. Big Data e Integração de Dados

Um dos obstáculos encontrados para construir uma cidade inteligente, ou qualquer outra aplicação de IoT, é o processamento e análise do grande volume de dados gerado. Atualmente a quantidade de dados produzidos diariamente pela humanidade é superior à

nossa capacidade de compreendê-los. Esses dados são gerados a partir de diversas aplicações, que vão desde atividades de usuários em redes sociais à sensores de dados climáticos. Mesmo considerando que uma grande quantidade de dados gerados não é importante e, portanto, não precisam de processamento, ainda assim restam enormes volumes de dados que precisam ser processados para gerar novos conhecimentos.

Além do desafio do processamento de um grande volume de dados, existe também o desafio de o fazer de forma eficiente e a um custo aceitável. O método tradicional é utilizar máquinas cada vez mais poderosas e armazenar os dados em bancos de dados relacionais para serem processados com a utilização de SQL. Porém, quando confrontado com dados típicos de cidades inteligentes, essas plataformas não são nem escaláveis nem adequadas. Novos modelos de processamento distribuído que vão além do tradicional SQL estão sendo desenvolvidos, sob o nome genérico de NoSQL [Leavitt, 2010]. Esse problema de pesquisa é atualmente chamado de Big Data [Adams, 2009], onde uma das grandes soluções atuais é o modelo MapReduce [Dean e Ghemawat, 2008], utilizado por grandes empresas como Google e Facebook [Thusoo, 2010]. Ele está disponível para os usuários através do Hadoop, um software livre mantido pela organização Apache.

Mais formalmente, Big Data é uma coleção de dados tão grande e complexa que se torna muito difícil processá-la usando ferramentas comuns de bancos de dados. Existem sete "Vs" que são associados ao conceito de Big Data:

1) **Volume**: Escala é certamente uma parte do que faz Big Data grande. A revolução da internet móvel, trazendo consigo uma enxurrada de atualizações de mídias sociais, dados de sensores de dispositivos e uma explosão de e-commerce, significa que todos os setores são inundados com dados que podem ser extremamente valiosos, se você souber como usá-los.

2) **Velocidade**: Refere-se ao aumento da velocidade com que esses dados são criados, e ao aumento da velocidade na qual os dados podem ser processados, armazenados e analisados pelos bancos de dados relacionais. As possibilidades de processamento de dados em tempo real é uma área que permite às empresas fazer coisas como anúncios personalizados de exibição nas páginas da web que visitadas, com base na pesquisa recente, visualização e histórico de compras do usuário.

3) **Variedade**: 90% dos dados gerados são "desestruturados", vindos de diversas fontes e formas possíveis. Eles podem ser dados de GPS, *tweets* com conteúdo e sentimento, dados visuais como fotos e vídeos, entre outras coisas.

4) **Variabilidade**: refere-se a dados cujo significado está em constante mutação. Este é especialmente o caso quando a coleta de dados se baseia em processamento de linguagem. As palavras não têm definições estáticas, e seu significado pode variar muito de acordo com o contexto.

5) **Veracidade**: Os dados são praticamente inúteis se não forem precisos. Isto é particularmente verdadeiro em programas que envolvem tomadas de decisões automatizadas ou algoritmos de aprendizado de máquina sem supervisão. Os resultados de tais programas são tão bons quanto os dados que eles estão trabalhando.

6) **Visualização**: Uma vez processados, é preciso encontrar uma maneira de apresentar os dados de forma legível e acessível. A visualização pode conter dezenas de variáveis e parâmetros, que vão muito além das variáveis x e y dos gráficos de barra padrão, e

encontrar uma maneira clara de apresentar esta informação é um dos desafios do Big Data.

7) **Valor:** O valor potencial do Big Data é enorme. Em essência, os dados por si só são praticamente inúteis. O valor encontra-se na análise rigorosa de dados precisos e nas informações/percepções que ela proporciona.

Muitos governos estão pensando em adotar o conceito de cidade inteligente em suas cidades e implementar aplicações de big data que suportem componentes de uma cidade inteligente para atingir o nível exigido de sustentabilidade e melhorar os padrões de vida de seus cidadãos. Cidades inteligentes utilizam várias tecnologias para melhorar o desempenho da saúde, transporte, energia, serviços de educação e de água, levando a maiores níveis de conforto de seus cidadãos. Isto envolve a redução de custos e consumo de recursos, além de mais eficácia e envolvimento ativo dos seus cidadãos. A análise e utilização do big data é uma das recentes tecnologias que tem um enorme potencial para melhorar os serviços de uma cidade inteligente. Como a digitalização tem se tornado parte integrante da vida cotidiana, a coleta de dados resultou no acúmulo de enormes quantidades de dados, que podem ser usados em vários domínios de aplicações benéficas. A análise e utilização de big data é o fator chave para o sucesso em muitos domínios de negócios e serviços, incluindo o domínio de cidade inteligente [Al Nuaimi *et al*. 2015].

Utilizar big data em uma cidade inteligente é indispensável em muitos cenários, como o transporte público, saúde, monitoramento de clima, segurança física, administrações governamentais, mobilidade, entre outros. Em uma cidade inteligente, a produção, gestão e difusão do conhecimento são essenciais, bem como o gerenciamento das próprias cidades e dos dados a ela relacionados. Sendo assim, cidades inteligentes e big data são as duas faces de uma mesma moeda. Não se pode trabalhar com um sem considerar o outro. Mas, apesar de indispensável, aplicar big data nesses cenários apresentam alguns desafios, como: (1) a grande variedade de fontes de dados; (2) a grande variedade de formatos dos dados; (3) custo para implantação; (4) a grande variedade de qualidade dos dados; e (5) adaptação e aceitação da população;

Além disso, questões como segurança e privacidade são fundamentais para o sucesso de uma aplicação de big data em uma cidade inteligente. Em termos básicos isso significa que os bancos de dados podem incluir informações confidenciais relacionadas com o governo e ao povo, então eles precisam de altos níveis de políticas e mecanismos de segurança para proteger esses dados contra uso não autorizado e ataques maliciosos. Além disso, aplicativos integrados com diferentes órgãos também exigem alta segurança, uma vez que os dados serão transmitidos através de vários tipos de redes, que podem ser seguras, ou não.

O que torna essa questão mais complexa é que a maioria das tecnologias de big data hoje, incluindo Cassandra e Hadoop, sofrem com a falta de segurança [Kim, Trimi e Chung, 2014]. Além da necessidade de proteger os dados à medida que eles trafegam na rede e quando eles estão sendo usados pelos diversos componentes de aplicações da cidade inteligente, há também a necessidade de identificar claramente e proteger os direitos de privacidade de organizações e indivíduos aos quais esses dados pertencem. Embora as entidades específicas da cidade inteligente possam reivindicar a posse da maior parte dos dados gerados pela análise do big data, a origem de tudo isso inclui informações pessoais e particulares sobre indivíduos. Registros médicos e de saúde,

registros financeiros e bancários, histórico de consumo, e muito mais, todos retratando características particulares das pessoas que eles representam. O acesso a esse tipo de dado é uma violação aos direitos legais à privacidade de um indivíduo. Certificar-se de que as políticas de privacidade são rigorosamente postas em prática e devidamente aplicadas, representa um grande desafio desenvolvedores e usuários de aplicativos para cidades inteligentes.

Para finalizar, é importante enfatizar que o monitoramento das cidades deve ser realizado em sintonia com a necessidade, o direito e o desejo dos cidadãos de terem sua privacidade mantida. É óbvio que um maior nível de monitoramento e análise de dados gera serviços mais precisos e eficientes, mas pode invadir o cidadão em situações indesejáveis. A coletividade, através da participação política e ativa, deve decidir os limites e barreiras às quais a introdução dessas novas tecnologias deve estar sujeita e quais os benefícios que podem ser auferidos.

2.3.6. Colaboração e Sensoriamento em Massa (crowdsourcing e crowdsensing)

A participação de um grande número de pessoas na resolução de tarefas ou detecção e comunicação de eventos, contextos e situações está se tornando cada vez mais comum, principalmente num ambiente altamente conectado onde o uso de *smartphones* é muito difundido. Alguns serviços de computação urbana já existentes tem demonstrado que o uso de colaboração em massa (*crowdsourcing*) e o sensoriamento em massa (*crowdsensing*) são técnicas que podem trazer grande contribuição para tornar as cidades mais inteligentes.

Para quem convive diariamente com o trânsito intenso em grandes centros urbanos, o uso do serviço Waze[8] já se tornou rotineiro, pela sua capacidade de economizar o tempo e diminuir os incômodos dos seus usuários. Sem a participação de um grande número de usuários, o Waze teria a utilidade de um GPS normal. Com a participação dos usuários, é possível saber em quais vias existe um maior acúmulo de veículos, em que velocidade os veículos estão trafegando nelas e onde e quando existem perigos ou pontos de atenção no caminho, como buracos ou carros no acostamento. Ao ligar o Waze, os usuários já estão enviando informações de localização, que ajudam os algoritmos a calcularem com maior precisão as melhores rotas. Além disso, os usuários podem colaborar explicitamente, indicando os locais onde ocorrem certos eventos.

Não há uma definição clara de *crowdsourcing*, mas em geral envolve uma tarefa complexa realizada coletivamente por um grupo de indivíduos em benefício de outros indivíduos ou organizações, que envolve benefício mútuo [Estellés-Arolas e González-Ladrón-de-Guevara 2012]. Quem lança a tarefa a ser executada tem a sua realização como benefício e quem realiza a tarefa pode fazê-lo por benefícios financeiros, reconhecimento social ou retribuição por outro serviço recebido. Esse termo tinha uma conotação mais relacionada com o um serviço prestado mediante pagamento há alguns anos atrás mas após a disseminação do uso de *smartphones* que estão sempre conectados (como nas cidades), outras formas de colaboração via *crowdsourcing* tem surgido, inclusive no contexto de cidades inteligentes.

Outra maneira de coletar dados em grandes cidades de maneira escalável é o uso do sensoriamento em massa (*crowdsensing*), onde o próprio indivíduo se torna um sensor,

[8] www.waze.com

como no caso do Waze. Enquanto que o *crowdsourcing* tenta realizar tarefas complexas dividindo-as em tarefas menores e atribuindo a sua solução a múltiplos indivíduos, no crowdsensing a responsabilidade da coleta das informações é da própria multidão [Cardone *et al.* 2013]. A técnica do *crowdsensing* pode trazer vários benefícios mas também novos desafios quando comparada às técnicas de sensoriamento urbano baseadas em infraestrutura, como os sensores em um ambiente de Internet das Coisas. Custo e facilidade de implantação são dois benefícios, enquanto que entre os desafios estão a imprevisibilidade da existência de indivíduos em certos locais para a obtenção de dados confiáveis. Em geral, a maioria dos pesquisadores acredita que *crowdsensing* é uma técnica complementar ao sensoriamento usando múltiplos sensores fixos.

O sensoriamento em massa pode auxiliar as cidades no tratamento de várias questões, inclusive como lidar com grandes aglomerações de pessoas, uma área conhecida como gerenciamento de multidões (*crowd management*). Aglomerações ocorrem voluntariamente em várias situações, como apresentações culturais, atividades esportivas e manifestações, mas também involuntariamente como em congestionamentos de veículos. Com a obtenção de dados dos próprios indivíduos é possível apresentar soluções para situações inesperadas, como evacuar multidões em caso de ameaça iminente [Franke *et al.* 2015].

2.3.7. Segurança e Privacidade

A segurança é muito importante para que aplicações de cidades inteligentes sejam adotadas em larga escala, já que este cenário é bastante vulnerável a ataques. Vários problemas de segurança podem ocorrer em um cenário urbano. Em geral, as tecnologias vêm sendo usadas em cidades inteligentes sem os devidos testes de segurança, tornando estes ambientes suscetíveis de serem explorados por potenciais atacantes. Em muitos casos, nenhum mecanismo de segurança é previsto ou controles fracos são utilizados [Cerrudo 2015]. Os sistemas são vulneráveis a ataques de negação de serviço (*DoS – Denial of Service*) e muito outros ataques ainda desconhecidos. Um simples ataque a uma parte insegura da rede pode afetar todo o resto, devido à interdependência das aplicações. Planos de emergência contra ataques cibernéticos precisam ser previstos pelas cidades para que tenham uma resposta rápida em caso de ataques [Cerrudo 2015]. Atacantes podem forjar um falso alarme de incêndio em um edifício, causando pânico na população, ou ainda atacar medidores inteligentes de energia resultando em um blecaute de um bairro ou mesmo de uma cidade inteira.

Soluções com altos níveis de confidencialidade, integridade e disponibilidade dos dados, bem como autenticação de usuários e dispositivos devem estar presentes nos serviços fornecidos. As questões de privacidade dos cidadãos também são essenciais, como quais dados serão fornecidos pelos usuários para cada serviço e como essas informações serão gerenciadas e acessadas. As pessoas também têm o direito de não fornecer dados para determinadas aplicações e saber dos riscos que correm se informações pessoais forem fornecidas a terceiros não autorizados. A questão é encontrar o equilíbrio entre privacidade e outros interesses e ter mecanismos e ferramentas que possam fazer estas tarefas. Assim, a escolha das tecnologias utilizadas deve considerar não só a funcionalidade, mas também as que possuam os controles adequados de segurança [Cerrudo et al 2015].

Vários requisitos básicos de segurança precisam ser tratados, de acordo com Cerrudo et al (2015). Esquemas robustos de criptografia precisam ser utilizados na

transmissão e armazenamento das informações. O uso e gerenciamento das chaves devem ser definidos, podendo em algumas aplicações serem utilizados certificados digitais e infraestruturas de chaves públicas (ICP). Os usuários e dispositivos precisam se autenticar antes da comunicação, seja com usuário e senha ou métodos mais sofisticados com certificados ou biometria. Antes de uma aplicação tomar qualquer ação, precisa ter as devidas permissões. Protocolos padronizados abertos como TLS (*Transport Layer Security*) e DTLS (*Datagram Transport Layer Security*) podem ser usados nas transmissões seguras, assim como protocolos de autenticação e autorização como OAuth, OpenID e OASIS SAML (*Security Assertion Markup Language*). A escolha dos protocolos adequados depende das limitações dos dispositivos, que podem ou não implementar mecanismos mais robustos.

Outro aspecto importante é a atualização automática de software e firmware. Com novas vulnerabilidades sendo descobertas a cada dia, esta atualização feita de maneira segura se torna necessária [Cerrudo et al 2015]. Os sistemas precisam ter controles para auditoria, alertas e logs. Os sistemas devem continuar seguros mesmo em casos de mau funcionamento (*fail-safe*). Em geral, devem ter um mínimo de controles de segurança configurados como padrão. Como muitos dispositivos podem ser acessíveis fisicamente por terceiros, devem possuir mecanismos antiviolação (*tamper proof*). Por exemplo, um usuário mal-intencionado pode ter acesso físico a um determinado sensor ou equipamento no carro, mas não pode adulterar a informação do dispositivo ou uma chave criptográfica. Uma forma de garantir que as soluções inteligentes tenham controles de segurança adequados é fazer com que os vendedores de soluções façam SLAs (*Service Level Agreements*) com as características de segurança do produto ou serviço [Cerrudo et al 2015]. As cidades podem criar CERTs (Centro de Estudos para Resposta e Tratamento de Incidentes) para lidar com incidentes de segurança e relatórios de vulnerabilidades, além de fazer testes de penetração (*pen tests*) regulares em toda a rede e serviços.

2.3.8. Computação Sensível ao Contexto

Na última década, computação sensível ao contexto foi amadurecendo e agora é considerada um componente importante para a Internet das Coisas [Perera *et al.* 2014]. Sistemas sensíveis ao contexto são capazes de adaptar seu comportamento às condições atuais, sem intervenção explícita do usuário [Baldauf *et al.* 2007]. O contexto geralmente refere-se a localização, mas pode compreender diferentes informações usadas para caracterizar entidades envolvidas na interação do usuário com a aplicação.

Um modelo de contexto é necessário para definir, manusear e armazenar as correlações entre as entidades que identificam o contexto. Existem diferentes modelos de contexto, como o orientado a objetos e o modelo baseado em ontologia [Makris *et al.* 2013]. A abordagem orientada a objetos estende esse paradigma para a modelagem de contexto. Modelagem baseada em ontologia organiza a informação em ontologias, utilizando abordagens semânticas, como a *Web Ontology Language* (OWL).

A inferência (*reasoning*) do contexto refere-se à informação ou ao conhecimento que pode ser inferido a partir da análise de dados e combinado com informações diferentes [Makris *et al.* 2013]. O processo de inferência do contexto pode ser dividido em três fases principais: o pré-processamento, a fusão de dados e a inferência. O pré-processamento é necessário para adaptar as informações de contexto de baixo nível a um formato adequado. A fusão de dados combina dados brutos de diferentes sensores

para extrair informações mais significativas. A inferência do contexto é o processamento de dados para extrair informações de alto nível, por diferentes abordagens, tais como inferência baseada em regras e inferência baseada em ontologia.

2.4. Cidades Inteligentes pelo Mundo

Várias cidades no mundo estão dando os seus primeiros passos em direção à oferta de serviços inteligentes e também estão sendo iniciadas experiências de criar cidades novas (do zero) que já usam uma grande variedade de tecnologias da computação urbana.

2.4.1. Santander (Espanha)

Com o objetivo de proporcionar um ambiente de experimentação de IoT em larga escala, de fato do tamanho de uma cidade, o projeto SmartSantander, na cidade de Santander na Espanha, conta aproximadamente 3000 dispositivos IEEE 802.15.4, 200 módulos GPRS e 2000 etiquetas inteligentes, entre RFID e QR Code. Este conjunto de dispositivos fornece informações ambientais, como temperatura, gás CO, ruído e luminosidade, rastreamento de vagas de trânsito, movimentação de veículos de transporte público, tráfego, irrigação de parques e jardins, além de permitir a intervenção dos usuários através de mecanismos de *crowdsensing*.

Um exemplo de experimento utilizando os dados fornecidos por esta infraestrutura é o monitoramento do transporte público (Figura 2.4), onde é possível fornecer em tempo real informações ao usuário deste transporte com alertas indicado que o ônibus está chegando ao ponto de embarque escolhido pelo usuário.

Esta mesma rede de sensores permite obter informações sobre o tráfego urbano, indicando melhores horários para locomoção dos cidadãos. As vagas de estacionamento público são monitoradas utilizando sensores ferromagnéticos permitindo informar em tempo real a disponibilidade de vagas aos motoristas [Sanches et.al, 2014]. Sensores de umidade presentes em parques e jardins permitem a irrigação precisa, considerando a demanda exata de água, precipitação passada e previsão futura, otimizando o uso do recurso natural.

Além da atuação instantânea, este grade volume de informações fornecidas pelo ambiente e seu histórico, permite aos administradores da cidade de Santander criar e modificar as políticas públicas baseados nas demandas e tendências apuradas. Apesar de alguns benefícios diretos desta plataforma já estarem em uso, os responsáveis pelo projeto afirmam que o potencial deste verdadeiro laboratório real, ainda está longe de ser totalmente explorado, e a expansão da rede de sensores e dispositivos apresenta um grande número de desafios tanto de infraestrutura como de processamento de dados.

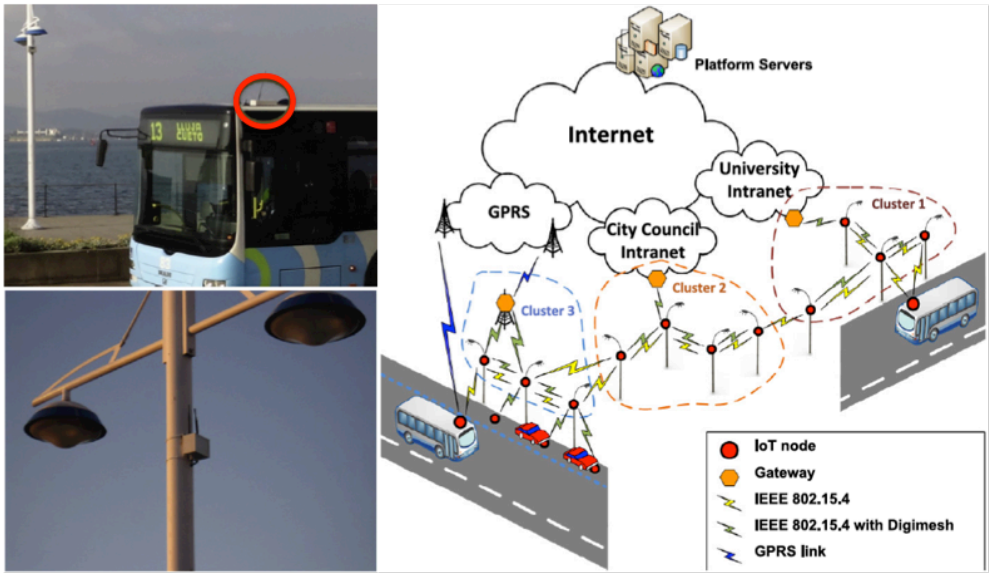


Figura 2.4. Exemplo de Aplicação em Transporte Público. Fonte [Sanches *et al.*, 2014].

2.4.2. Pádua (Itália)

O projeto *Padova Smart City* [Cenedese *et al.* 2014], desenvolvido com a colaboração da iniciativa pública e privada além da Universidade di Padova, é uma implementação prática de tecnologias de IoT aplicadas na cidade de Pádua na Itália. Com dispositivos acoplados aos postes de iluminação pública da cidade, sensores de luz, CO, temperatura, umidade, vibração e ruído dispostos por toda a cidade e interligados por uma rede sem fio 802.15.4 (ZigBee), utilizando o protocolo CoAP sobre UDP para o transporte das informações. A Figura 2.5 mostra um trecho da cidade de Pádua com seus respectivos postes de luz que possuem os coletores de dados.

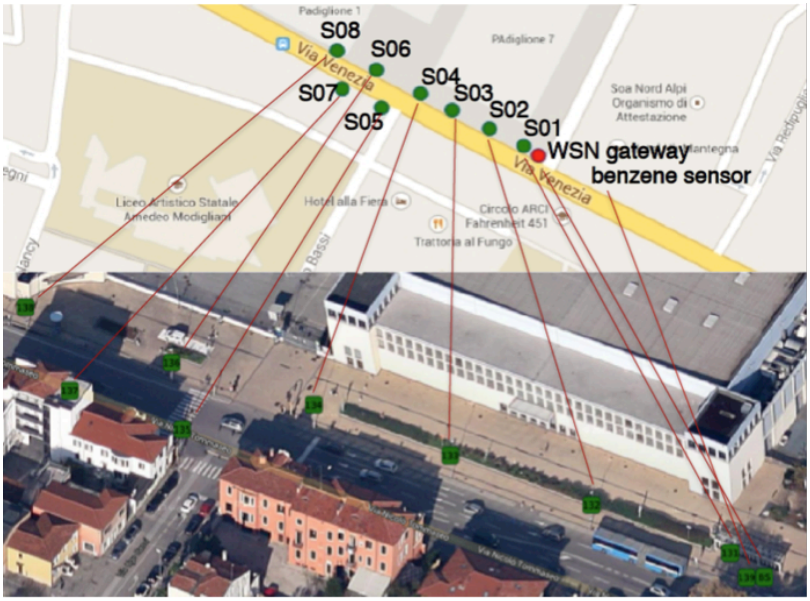


Figura 2.5. Trecho da Cidade de Pádua na Itália com os postes de coletores de dados. Fonte: [Cenedese *et al.* 2014]

As primeiras análises realizadas nos dados obtidos, apresentados em [Cenedese *et al.* 2014] tem como objetivo a compreensão dos fenômenos relacionados com a variação nos dados. Entre outras informações, foi possível observar o efeito de luzes de tráfego, sombras causadas por nuvens de chuva e até mudança de hábitos durante festas comemorativas. Este entendimento é fundamental para a produção de conhecimento sobre os dados obtidos em tempo real além de permitir a predição e atuação no ambiente da cidade.

2.4.3. Amsterdam (Holanda)

A iniciativa Amsterdam Smart City (ASC)[9] é o resultado da parceria entre iniciativa privada, governo, instituições de pesquisa e a população de Amsterdam, na Holanda, com o objetivo de transformá-la em uma verdadeira cidade inteligente. Fundada em 2009, a iniciativa ASC já conta com mais de 90 projetos envolvendo mais de 100 parceiros entre empresas e institutos de pesquisa, dividindo seus esforços nas áreas seguintes áreas:

Smart Mobility - o objetivo é prover um sistema de transporte variado, seguro, eficiente e confortável e conectado com a infraestrutura de telecomunicações. Projetos com objetivo de redução de emissão de poluentes, combustíveis alternativos, transporte utilizando bicicletas, estações de carga para veículos elétricos, gerenciamento de tráfego com sugestões de trajeto aos motoristas, compartilhamento de carros, entre muitas outras iniciativas fazem parte deste foco.

Smart Living –o foco é o local de moradia e de trabalho, no tempo relacionado com estas atividades, deslocamentos e no impacto na qualidade de vida. Aqui problemas como casas sustentáveis, armazenamento de energia em casa, sustentabilidade dos canais navegáveis de Amsterdam, climatização de casas, criação de escritórios compartilhados em prédios históricos, entre outros projetos.

Smart Society – nesta área o foco é no desenvolvimento pessoal, tanto de residentes como visitantes, incentivando a criatividade e a interação social. Entre os projetos, destaca-se a criação de laboratórios públicos com máquinas e equipamento para desenvolvimento de ideias e experimentos por qualquer cidadão, revitalização de áreas degradadas da cidade, plataformas para compartilhamento de ideias e projetos para a cidade, grupos de discussão on-line para encorajar o engajamento dos cidadãos a respeito dos problemas da cidade e redes de financiamento solidário.

Smart Areas - trata o problema do planejamento urbano ligado à sustentabilidade e uso eficiente de matérias. Entre as iniciativas, destaca-se a criação de plataformas para simulação 3D da cidade para simulações de novas ideias, projetos para reduzir a vulnerabilidade da cidade às chuvas, criação de distritos industriais, iluminação pública flexível, instalação painéis fotovoltaicos em tetos de grandes construções e aproveitamento do calor gerado por resíduos industriais.

Smart Economy - tem por objetivo tornar a cidade atraente para o empreendedorismo, estimulando a inovação e atraindo a atenção de investidores internacionais. Projetos visando a participação ativa dos cidadãos das políticas públicas, incentivando parceiros na construção da iniciativa ASC como um todo e centros de tele-presença fazem parte desta área.

[9] http://amsterdamsmartcity.com/

Big & Open Data – os dados são o combustível da sociedade inteligente e o grande volume de dados um grande desafio. Esta área propõe não só padrões abertos e incentivos ao compartilhamento destes dados, como também a discussão sobre o que deve ser protegido e o que deve ser divulgado. Entre os projetos desenvolvidos nesta área destaca-se a criação do CitySDK (*CityService Development Kit*) não só para simplificar o desenvolvimento de aplicações para cidades inteligentes com também definindo formatos e padrões para tornar estes dados públicos.

Infrastructure – Todos este projetos demandam uma infraestrutura de telecomunicações capaz de suprir as demandas geradas pela cidade. Entre os projetos desta área, destacam-se o *Amsterdam Free Wifi*, combinando uma rede de fibras óticas e antenas por toda a cidade e a distribuição de fibras óticas para interligar as casas com o projeto *Fiber-to-the-home*.

Living Labs – Fornecer um ambiente que incentiva a criatividade e a possibilidade de implementar novas ideias em um ambiente real são características de uma sociedade inteligente. Entre os projetos abraçados por esta área, destaca-se o incentivo aos estudantes implementarem novas ideias e obter a resposta imediata de seu impacto junto aos cidadão, criação de jogos para envolver os residentes nas soluções propostas pelos projetos desenvolvidos, permitindo a adoção ampla destas iniciativas.

Um dos projetos que integram a iniciativa ASC é o Smart Citizen Kit (Figura 2.6), um coletor de dados ambientais, que os residentes podem adquirir por aproximadamente €160, para fornecer dados em tempo real de temperatura, umidade, ruído, níveis de CO e NO_2 e luminosidade.

Figura 2.6. Smart Citizen Kit[10].

Este Kit é desenvolvido utilizando a arquitetura Arduino[11] e conta com diversos sensores, uma bateria, rede WiFi e conexão permitindo o uso de painéis solares para

[10] http://docs.smartcitizen.me/#/start/hardware

uma operação totalmente autônoma. Uma vez instalado, o Kit permite acesso via internet ao dados em tempo real ou através de um aplicativo para smartphones.

2.4.4. Chicago (EUA)

Chicago é a cidade de grande porte nos EUA que está avançando mais rápido rumo a oferecer serviços urbanos mais inteligentes a seus cidadãos. É a primeira cidade que está construindo uma infraestrutura permanente para coletar dados provenientes de uma grande quantidade de sensores através do projeto *Array of Things* (AoT)[12], uma parceira da Universidade de Chicago, com o Argonne National Laboratory e a prefeitura de Chicago. Outra iniciativa é Chicago Beta City[13], um plano para tornar a cidade mais atrativa para empresas de tecnologia, além de prepará-la para lidar com as tecnologias futuras. Uma primeira ação dentro da Chicago Beta City é a criação de um ambiente para testar carros sem motorista, como o carro auto dirigível do Google[14], em áreas especiais da cidade, chamadas de "zonas de inovação".

Além disso, a cidade aderiu à iniciativa de dados abertos e através do Portal de Dados de Chicago[15] que permite o acesso a mais de 200 conjuntos de dados em áreas variadas como administração e finanças, educação, ambiente e desenvolvimento sustentável, eventos, saúde, segurança pública e transportes. O objetivo é dar transparência aos dados e ao mesmo tempo permitir o desenvolvimento de ferramentas que forneçam serviços avançados aos cidadãos e à municipalidade através do uso desses dados. Por último, Chicago está implantando uma infraestrutura para dar suporte à uma rede elétrica inteligente (Smart Grid), com a instalação de medidores inteligentes em toda a cidade[16].

Array of Things (AoT) é um projeto de sensoriamento urbano consistindo de uma rede de nós com sensores de diferentes naturezas embutidos em uma única caixa que estão sendo instalados em postes na região central de Chicago para coletar dados ambientais, de infraestrutura e atividade para uso em pesquisas e serviços ao público. Os dados coletados dos sensores serão disponibilizados no portal de dados de Chicago para permitir o desenvolvimento de aplicações e serviços variados pela municipalidade, empresas, grupos de software livre e pesquisadores. O objetivo é que esses dados auxiliem Chicago a dar mais um passo para se tornar uma legítima cidade inteligente, funcionando de maneira mais eficiente, reduzindo custos e solucionando problemas de maneira proativa. A Figura 2.7 apresenta a arquitetura do AoT com seus principais componentes.

[11] http://www.arduino.org/
[12] https://arrayofthings.github.io
[13] http://chicagoinno.streetwise.co/topic/beta-city-initiative
[14] https://www.google.com/selfdrivingcar
[15] https://data.cityofchicago.org
[16] http://www.cityofchicago.org/city/en/progs/env/smart-grid-for-a-smart-chicago.html

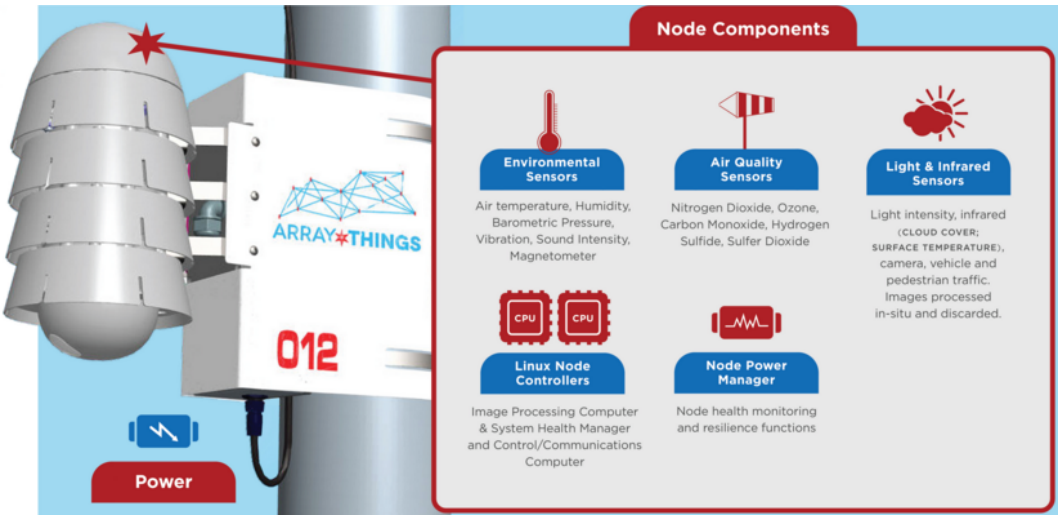


Figura 2.7. Componentes do Array of Things (AoT). Fonte: arrayofthings.github.io

Inicialmente os nós do AoT irão medir temperatura, pressão, luminosidade, vibração, monóxido de carbono, dióxido de nitrogênio, dióxido de enxofre, ozônio, intensidade de som ambiental e tráfego de veículos e pedestres. Está previsto o desenvolvimento futuro de novos sensores para mensurar outros fatores de interesse como níveis de chuva e inundação, vento e poluentes. O projeto AoT tem uma preocupação com a manutenção da privacidade dos cidadãos, de modo que um requisito importante no projeto dos sensores foi um bloqueio para impedir a coleta de dados individuais.

2.4.5. São Paulo (Brasil)

São Paulo é uma das cidades brasileiras que vem desenvolvendo iniciativas mais ambiciosas e efetivas para se tornar mais inteligente e oferecer melhor qualidade de vida aos cidadãos. Para que os conceitos de cidades inteligentes sejam implementados e se tornem realidade, existem necessidades mais básicas que devem estar disponíveis a todos os cidadãos em todos os momentos. A principal delas é o acesso universalizado à conectividade da rede. Conectividade é essencial para viabilizar a construção de cidades mais inteligentes, para que "coisas", pessoas, servidores, aplicações e dispositivos variados possam todos estar integrados de modo a contribuir para gerar serviços diversos ao cidadãos e à municipalidade.

Em 2014 a Prefeitura de São Paulo iniciou a implantação do programa WiFi Livre SP[17], desenvolvido com o objetivo de tornar a Internet mais acessível ao cidadão, disponibilizando sinal WiFi livre e gratuito em praças, parques e outros locais públicos. O programa WiFi Livre SP implantou 120 praças digitais, nas cinco regiões da cidade: Centro, Norte, Sul, Leste e Oeste. Nestas praças digitais o acesso é irrestrito e gratuito a qualquer cidadão, que pode fazer uso da rede por meio de diferentes dispositivos, como notebooks, tablets ou smartphones. A Figura 2.8 apresenta um mapa da localização das praças digitais com WiFi gratuito em São Paulo.

[17] http://wifilivre.sp.gov.br

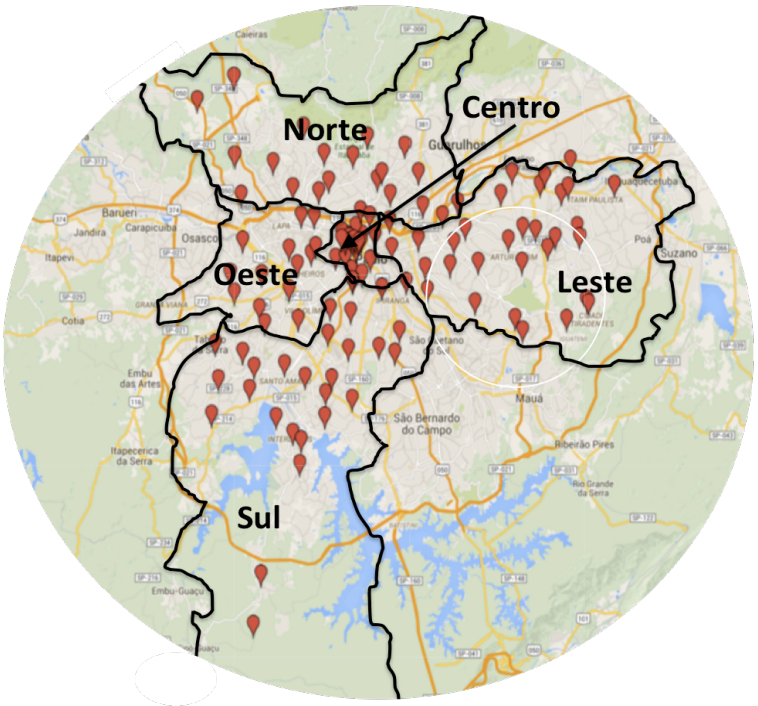


Figura 2.8. Praças Digitais em São Paulo com WiFi Gratuito

Existem diferentes modelos de implantação de redes WiFi de acesso gratuito em uma municipalidade, que podem variar de situações onde o poder público é proprietário de toda a infraestrutura de rede ou usa conexões alugadas a modelos onde uma ou várias empresas privadas realizam o serviço mediante um contrato. Independente do modelo utilizado, sempre haverá a necessidade de mensurar e divulgar métricas de desempenho da rede que demonstram a qualidade e estabilidade da conexão oferecida à Internet. No caso do programa WiFi Livre SP foi adotado o modelo de contratação de empresas para prestação do serviço de conectividade.

Para aferir os resultados da política pública de abertura gratuita de sinal WiFi a Prefeitura de São Paulo e a Universidade Federal do ABC (UFABC) estabeleceram uma parceria através do projeto "Conectividade e Inclusão Digital para São Paulo". Esta parceria visa estudar os efeitos da Internet, em suas múltiplas dimensões, na vida dos cidadãos e das comunidades atendidas, com a finalidade de gerar dados para fomentar a implementação e avaliação de uma política pública de acesso gratuito a Internet por meio de rede sem fio. Os estudos e dados obtidos são importantes ferramentas para a gestão da política de conectividade e inclusão digital, que além de medir os efeitos da política. Também permitem realizar os ajustes e correções necessárias, condições necessárias para uma maior efetividade e eficácia da ação, como também auxiliar no desenvolvimento de novos projetos voltados para o aprofundamento da inclusão digital e promoção da cidadania.

Além da conectividade, a cidade de São Paulo está trabalhando para que todas as 700 mil lâmpadas públicas sejam trocadas por lâmpadas de LED e neste processo todas elas sejam equipadas com sensores e atuadores. Além da economia imediata de cerca de 50% proporcionado pela tecnologia LED, os dados coletados das lâmpadas podem ser

analisados para que novas economias possam ser geradas. Esse é um passo efetivo para que seja possível equipar São Paulo com iluminação inteligente.

2.4.6. Criando Cidades Inteligentes do Zero

Construir cidades inteligentes totalmente novas não é algo possível para a grande maioria das cidades do mundo, mas existem algumas iniciativas caminhando neste sentido, como a nova cidade de Songdo na Coreia do Sul e a cidade de testes CITE City nos EUA.

SongDo - Coreia do Sul

A cidade de Songdo[18] na Coreia do Sul está sendo construída como uma cidade inteligente do futuro, em uma área aterrada de 600 hectares próxima a Seul que antes pertencia ao mar. O nome oficial é *Songdo International Business District* (Songdo IBD) e está se tornando um modelo de desenvolvimento de novas cidades inteligentes ao custo de 35 bilhões de dólares. Quanto estiver completa, com previsão agora revisada para 2020, Songdo terá 80 mil apartamentos, 50 mil metros quadrados de espaços comerciais, um hospital, um centro de convenções, um complexo acadêmico e muito espaço verde. Todos os cidadãos e empresas terão acesso em tempo real a uma quantidade de informações sobre a cidade, uma vez que tudo estará interligado. Sensores controlam muitos aspectos da vida na cidade e a telepresença é praticamente onipresente.

Uma rede WAN conecta as estruturas presentes em Songdo para oferecer uma grande variedade de serviços digitais. Por exemplo, apartamentos tem painéis para controlar a iluminação, temperatura e acesso a informação. Está previsto que pelo menos um quarto dos apartamentos poderão usufruir de tecnologias de telepresença. Um datacenter verde (*green datacenter*) gerencia todos os aspectos da vida urbana, incluindo controle de trânsito, fornecimento de água, gás e energia e reciclagem. A Cisco é a principal empresa fornecedora de tecnologia para Songdo através da sua inciativa Smart+Connected Communities[19].

CITE City - EUA

No deserto do estado do Novo México nos Estados Unidos está sendo construída uma cidade para aproximadamente 35 mil habitantes ao custo de 1 bilhão de dólares onde ninguém irá morar. Ela é chamada CITE[20] (*Center for Innovation, Testing and Evaluation*) e está sendo construída pela empresa Pegasus Global Holdings para ser usada como laboratório de testes em proporções reais para viabilizar pesquisa, desenvolvimento, inovação e comercialização de novas tecnologias urbanas. CITE será usada como uma etapa intermediária para novas tecnologias, entre os testes de laboratório e abertura para o público geral. Por exemplo: a) carros autônomos poderão ser testados em ruas e estradas adaptadas com tecnologias reativas sem perigo de causar acidentes que coloquem vidas de pessoas em risco; b) casas totalmente automatizadas e com sistemas robóticos poderão ser projetadas e testadas; c) fontes de energia

[18] http://songdoibd.com

[19] http://www.cisco.com/c/en/us/solutions/industries/smart-connected-communities.html

[20] http://www.cite-city.com

alternativa poderão ser testadas em larga escala. Entre os usuários de CITE, a Pegasus espera atrair a atenção de laboratórios e agências do governo dos EUA em áreas como energia, transporte, segurança da informação, cidades inteligentes, além de universidades e centros de pesquisa, indústrias e concessionárias de serviços públicos.

Para possibilitar a realização de uma grande variedade de experimentos, a cidade será composta de vários laboratórios de campo (Field Labs), conforme mostra a Figura 2.9. O principal laboratório é o City Lab, que contém várias áreas típicas de uma cidade americana com cerca de 35 mil habitantes.

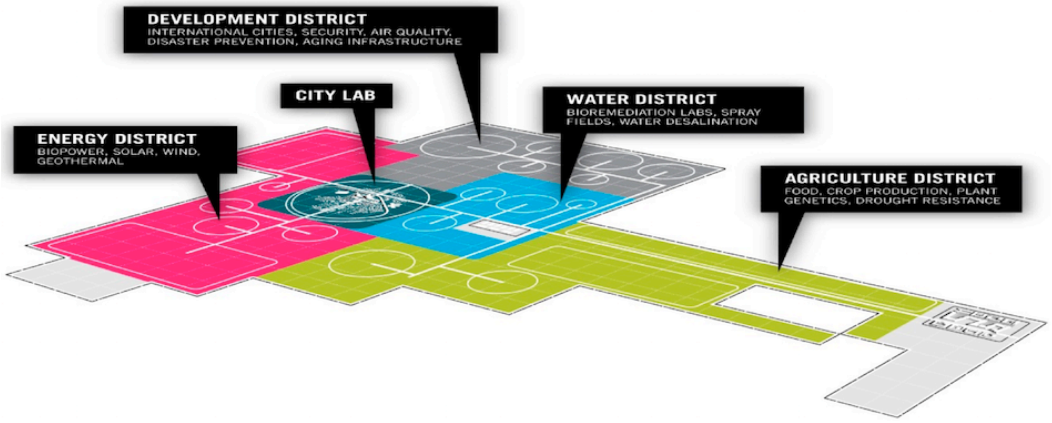


Figura 2.9. CITE Field Labs (fonte: www.cite-city.com/Facility/Main/CITE_Field_Labs.html)

Para gerenciar e monitorar uma cidade de testes automatizada, CITE será controlada por um sistema de comunicação e computação chamado de CITE Backbone. A cidade inteira será monitorada por sensores que enviarão dados a um Centro de Operação de Redes, suportado por um Data Center específico. O Backbone irá conter uma rede de túneis que se estende por toda a cidade para monitorar todas as funções da cidade, incluindo casas e prédios residenciais, áreas industriais, centros comerciais e centros de fornecimento de energia, água e gás e de coleta de resíduos.

Com essas características, CITE pode se transformar nos próximos anos no maior laboratório ao ar livre para novas tecnologias e conceitos de computação urbana para viabilizar cidades inteligentes.

2.5. Pesquisa em Soluções e Tecnologias para Cidades Inteligentes

Cidades inteligentes não necessitam de um grupo exclusivo de tecnologias, mas uma série de conceitos, plataformas computacionais e tecnologias que usados em conjunto podem viabilizar que vários novos serviços sejam oferecidos aos cidadãos de grandes espaços urbanos. Essa seção apresenta alguns resultados de pesquisa com tecnologias que focam em cenários de cidades inteligentes.

2.5.1. Gerenciamento Elástico de demanda em Smart Grid na Nuvem

Computação em nuvem necessita do gerenciamento de elasticidade para que recursos possam ser alocados e desalocados dinamicamente sob demanda. Embora a adoção de serviços de nuvem venha crescente, não é muito claro para os usuários como configurar a elasticidade em diversas situações diferentes. Essa seção apresenta resultados de avaliação de desempenho de elasticidade em um cenário de Smart Grid (Rede Elétrica

Inteligente) onde as negociações entre clientes e distribuidoras apresenta intensidades variáveis de acordo com os padrões diários de consumo de energia [Simões e Kamienski 2014]. Os experimentos consideram o uso de uma nuvem privada e híbrida que combina a nuvem privada com a nuvem pública da Amazon, usando o serviço EC2. Além disso, os clientes são alocados em nós do PlanetLab ao redor do mundo, que funciona como uma nuvem comunitária.

O cenário de avaliação de elasticidade é o sistema de gerenciamento de Smart Grid apresentado na Figura 2.10. O sistema é composto de três módulos principais: geração, distribuição e consumo de energia. A geração é um módulo simples que simula a quantidade de energia disponível para os clientes pela empresa responsável pela geração e transmissão, implementado em Java na infraestrutura de nuvem privada. O módulo de consumo gera pedidos de demanda de energia ao módulo de distribuição e é implementado no PlanetLab. Esses pedidos são enviados continuamente para que os clientes possam se beneficiar de preços diferenciados durante o dia. O módulo de distribuição é responsável por negociar pedidos de energia provenientes dos clientes que apresentam grande variação sazonal. Por esse motivo, ele é executado em uma infraestrutura de nuvem com gerenciamento de elasticidade, onde a carga de trabalho variável gera a criação e a destruição de máquinas virtuais (VMs). A nuvem privada usa o controlador de nuvem OpenNebula com o hipervisor Xen. O serviço de distribuição é replicado em múltiplas VMs através de um serviço de balanceamento de carga. Quando a capacidade da nuvem privada for esgotada, a carga de trabalho adicional é repassada à nuvem pública da Amazon através de um mecanismo chamado de *cloudburst* que cria uma nuvem híbrida.

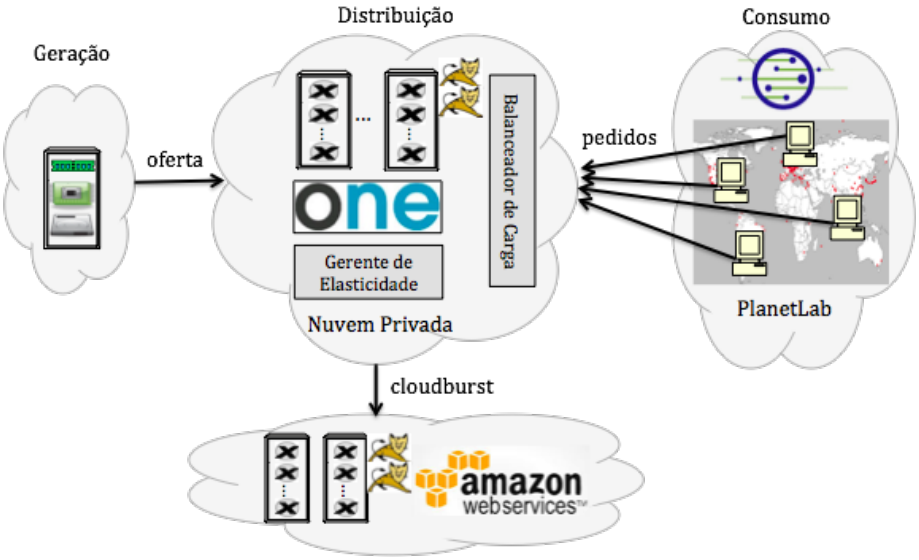


Figura 2.10. Cenário de Smart Grid baseado em nuvem computacional

A Figura 2.11 mostra resultados de experimentos com nuvem privada e híbrida em que a taxa de chegada de pedidos aumenta e diminui conforme o tempo de realização do experimento progride. A carga de trabalho foi elevada de zero até 3500 pedidos simultâneos e depois novamente diminuída até zero. A métrica usada para gerenciar a elasticidade é o tempo de resposta com uma política determinando um limiar superior e um inferior para criar e destruir uma VM respectivamente. A Figura 2.11(a) mostra a

série temporal para o tempo de resposta na nuvem privada, que visivelmente não é afetada pelo aumento da carga, devido ao aumento regular do número de VMs, como mostra a Figura 2.11(b). O SLA de tempo de resposta é definido com 10 segundos e é possível observar que a média e a mediana são mantidas abaixo de 3 segundos enquanto que o percentil 90 é mantido abaixo de 9 segundos. Por outro lado, quando a carga excessiva é direcionada para a nuvem pública da Amazon com instâncias (VMs) do tipo pequeno *(small)*, os tempos de resposta aumenta significativamente, como mostra a Figura 2.11(c) (VMs alocadas na nuvem privada e pública são mostradas na Figura 2.11(d)). Com o uso de instâncias do tipo médio *(medium)* e grande *(large)* o desempenho para nuvem híbrida foi semelhante ao desempenho em nuvem privada (resultado não mostrado na figura).

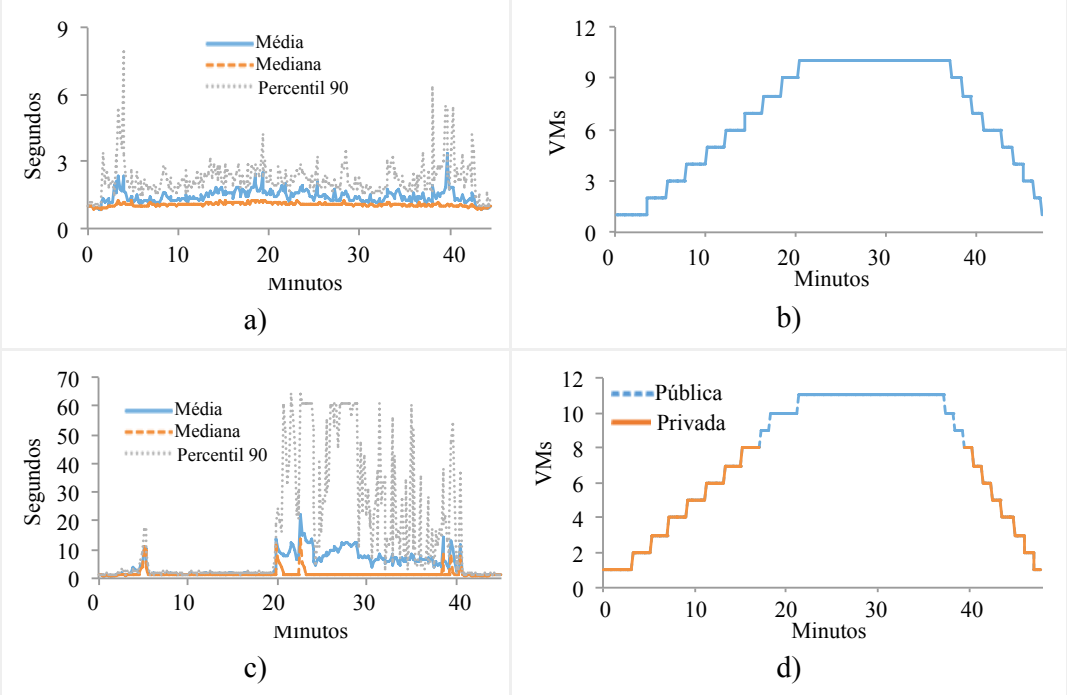


Figura 2.11. Resultados de elasticidade em nuvem computacional para Smart Grid

2.5.2. Conectividade: O impacto de uma rede WiFi pública

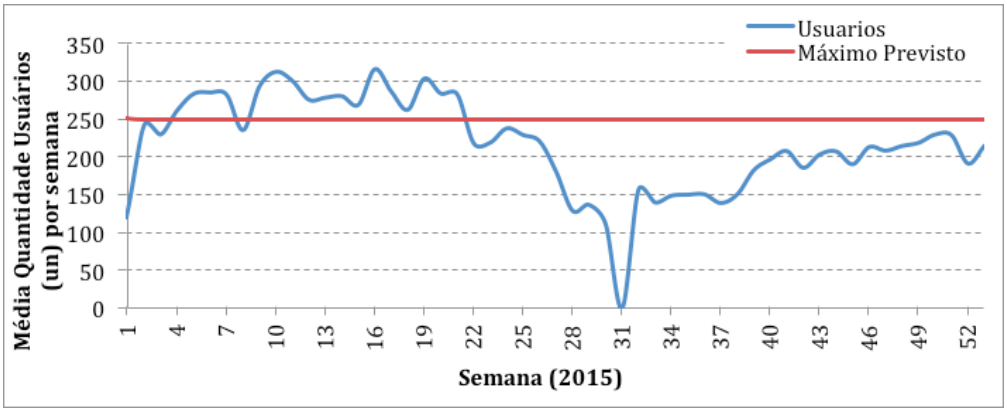
Em 2014 a Prefeitura de São Paulo iniciou a implantação do programa WiFi Livre SP[21], desenvolvido com o objetivo de tornar a Internet mais acessível ao cidadão, disponibilizando sinal WiFi livre e gratuito em praças, parques e outros locais públicos. O programa WiFi Livre SP implantou 120 praças digitais, em cinco regiões da cidade: Centro, Norte, Sul, Leste e Oeste. Nestas praças digitais o acesso é irrestrito e gratuito a qualquer cidadão, que pode fazer uso da rede por meio de diferentes dispositivos, como notebooks, tablets ou smartphones. A pesquisa descrita nessa seção faz parte de uma iniciativa que visa estudar os efeitos da Internet, em suas múltiplas dimensões, na vida dos cidadãos e das comunidades atendidas pela política de abertura de sinal de rede sem fio em São Paulo. Esses estudos têm a finalidade de gerar dados para fomentar a

[21] http://wifilivre.sp.gov.br

implementação e avaliação de uma política pública de acesso gratuito a Internet por meio de rede sem fio [Ratusznei *et al.* 2015].

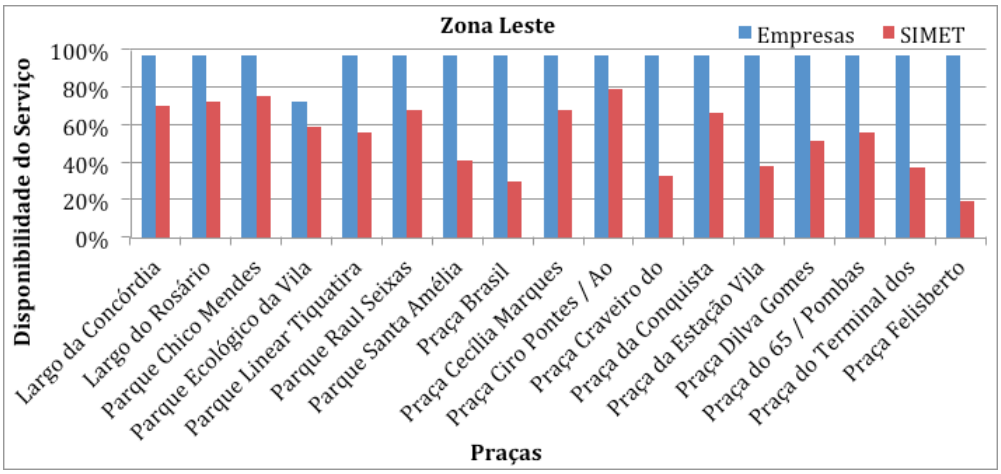
Foram coletados dados provenientes das empresas prestadoras de acesso à Internet e do serviço de medição SIMET[22] instalado em 112 praças que estiveram funcionando durante todo o ano de 2015. As avaliações se concentraram em três aspectos que relevam para o cidadão a qualidade da experiência (QoE) ao utilizar o serviço: acesso, disponibilidade e desempenho. O primeiro contato do usuário com o serviço de conectividade é através da conexão com a rede WiFi. Para avaliar o acesso são apresentados dados sobre o número de usuários conectados em cada praça, comparando-os com o número máximo preestabelecido para ela. As praças devem oferecer serviço contínuo aos usuários com alta disponibilidade. Medir a disponibilidade é importante para garantir que o acesso à Internet proporcionado pelas praças digitais atenda às demandas da população em todos os momentos que forem necessários. Por último, cada praça deve oferecer um serviço compatível com certos parâmetros de qualidade, como taxa de transferência (download) e latência (atraso na transmissão).

A Figura 2.12 mostra algumas amostras de resultados de número médio de acessos, disponibilidade e desempenho de praças digitais em São Paulo. A Figura 2.12(a) mostra a média semanal de usuários conectados no Mercadão Municipal, no centro de São Paulo. É possível observar que em boa parte do ano, o número médio semanal estava acima do número máximo de usuários previstos para essa praça. Ou seja, o número máximo foi usado apenas para o aprovisionamento de recursos nas praças, não sendo imposto um limite superior para barrar a conexão de novos usuários. A Figura 2.12(b) mostra a disponibilidade média do serviço nas praças da Zona Leste, medidos pelos dados disponibilizados pela empresa prestadora do serviço e pelo SIMET. É possível observar que pelos dados da empresa, as praças estavam disponíveis quase 100% do tempo, mas pelo SIMET a disponibilidade varia em torno de 50%. Essa diferença se deve tanto à indisponibilidade real das praças quanto a problemas com o serviço SIMET. Por último, a Figura 2.12(c) mostra as médias de número de usuários e taxa de entrada por usuário calculados para cada hora do dia para a Praça Zilda Natel na Zona Oeste. O comportamento é esperado, com um maior número de usuários durante o período diurno e consequente menor taxa de transferência disponível para cada usuário.

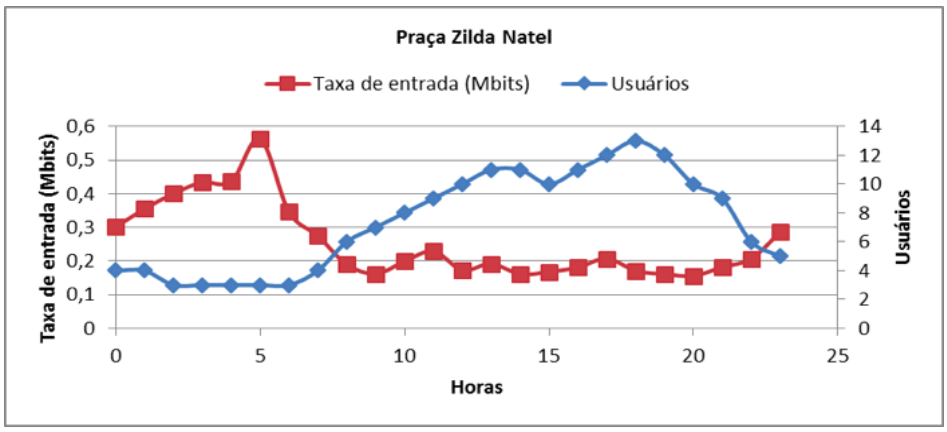


a) Número médio semanal de usuários no Mercadão Municipal

[22] http://simet.nic.br



b) Disponibilidade das Praças da Zona Leste



c) Taxa de entrada por usuário vs. número de usuários por hora
(Praça Zilda Natel - Zona Oeste)

Figura 2.12. Resultados de acesso, disponibilidade e desempenho de praças digitais em São Paulo

2.5.3. Prédios Inteligentes: Gerenciamento Sensível ao Contexto baseado em IoT

A geração e o uso eficiente de energia estão atualmente em evidência, pelo impacto natural que impõem às questões relacionadas ao uso sustentável dos recursos naturais. Como a demanda mundial por energia tende a crescer, surge o desafio de usar soluções tecnológicas para gerenciar seu uso de maneira mais eficiente. O caminho para transformar os espaços urbanos em verdadeiras cidades inteligentes encontra muitos desafios, entre os quais o uso eficiente de energia em prédios.

IoT desempenha um papel fundamental na implantação de cidades e prédios inteligentes, viabilizando a coleta de dados de uma enorme quantidade de sensores, a análise e tomada de decisões e o envio de comandos para atuadores para alterar o comportamento dos sistemas. A computação baseada em contexto oferece a inteligência necessária à IoT, junto com Big Data e outras tecnologias, para que os comportamentos possam ser adaptados na velocidade em que é necessário, sem a direta intervenção humana [Osello *et al.* 2013]. O desenvolvimento de sistemas baseados em IoT sensíveis

ao contexto ainda exige esforços tremendos, devido à falta de ferramentas para otimizar o processo. O projeto IMPReSS[23] tem o objetivo de desenvolver uma plataforma para o desenvolvimento rápido de sistemas IoT baseados em contexto, com foco em sistemas para uso eficiente de energia.

Essa seção apresenta o Gerenciador de Contexto do IMPReSS, um componente de middleware que permite adaptação sensível ao contexto à plataforma [Kamienski *et al.* 2015]. Ele simplifica atividades de identificação e modelagem de contexto, assim como oferece funções de gerenciamento, como fusão de dados, inferência de contexto e comunicação com atuadores para gerar alterações de comportamento. A inferência do contexto e tomada de decisão é baseada em regras. Um protótipo foi desenvolvido para controle de iluminação e temperatura em uma sala de aula de uma universidade, que foi usado para diversas finalidades, como experimentos, demonstrações, identificação de problemas e avaliação de desempenho. A experiência trabalhando com o protótipo mostra que o Gerenciador de Contexto permite a configuração de mudanças automáticas e rápidas do sistema sensível ao contexto, mesmo com um cenário simples da sala de aula. Foi realizado um estudo de avaliação de desempenho para identificar possíveis gargalos para a escalabilidade do sistema, que possibilitou compreender melhor o comportamentos dos seus componentes, principalmente dos responsáveis pela realização de fusão de dados (Esper) e do processamento de regras (Drools).

A Figura 2.13 mostra a arquitetura do Gerenciador de Contexto, com os seus principais componentes.

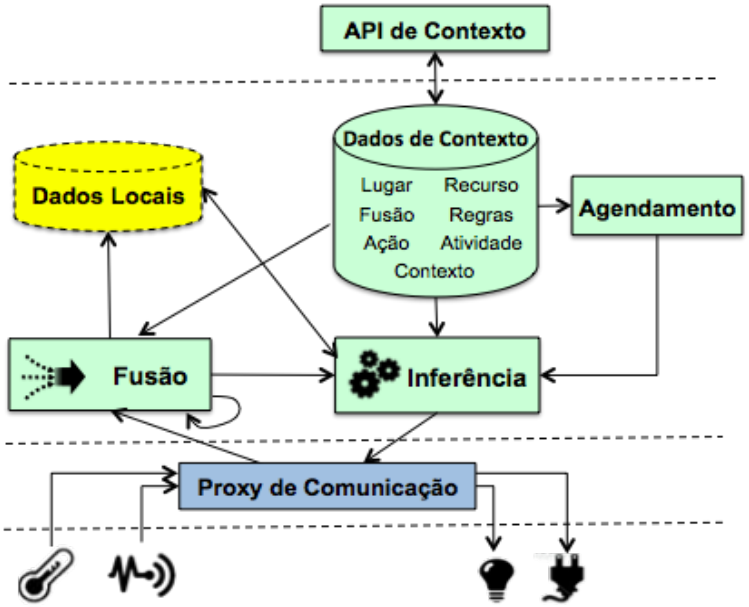


Figura 2.13. Gerenciador de Contexto

A API de Contexto é parte da API do IMPReSS e expõe uma interface REST, permitindo que outros módulos interajam com o Gerenciador de Contexto. O Armazenamento de Contexto realiza o armazenamento e recuperação de templates de entidade de contexto, por meio da API de contexto. Foi utilizado o sistema de

[23] http://impressproject.eu

mapeamento objeto-relacional (ORM) EclipseLink com PostgreSQL. A Fusão de dados utiliza de um conjunto de técnicas para combinar dados de múltiplas fontes (sensores) e é realizada pelo Esper, que pode executar Processamento de Eventos Complexos (CEP) por filtragem, configurável através de uma linguagem como SQL. O motor de Inferência (Drools) infere consequências lógicas de um conjunto de fatos, usando regras para processar os pedidos que chegam da Fusão e gerar ações enviadas aos atuadores. O Agendador gerencia a agenda de eventos, tais como aulas em uma universidade e dispara a Inferência para tomar as ações apropriadas. O Proxy de Comunicação encapsula a comunicação com os sensores e atuadores, interfaceando com o Gerenciador de Comunicação e com um broker MQTT.

A Figura 2.14 apresenta resultados de avaliação de desempenho do Gerenciador de Contexto, em um cenário de uma sala de aula em uma universidade, em que luzes são acesas e apagadas automaticamente de acordo com a presença de alunos. O número de sensores foi variado de 50, 200 e 600 e a fusão de dados calculou a média de 1, 3 ou 30 medições vinda dos sensores. Na Figura 2.14(a) é possível observar o efeito do aumento da carga de mensagens no sistema (número de sensores) e do número de dados para realização da fusão. É possível observar que o efeito da realização das fusões de dados no Esper é maior para um número menor de dados por fusão e carga de trabalho mais alta (número de sensores). Ou seja, o desempenho do Esper é afetado mais incisivamente pelo número de fusões realizadas do que pelo número de dados a serem usados na fusão. A comparação direta da Figura 2.14(a) com a Figura 2.14(b) revela que o tempo de processamento das regras não é majoritário no tempo total de processamento, na média representando menos de 10% do total. No entanto, é possível observar que há um aumento significativo no tempo de processamento das regras com o crescimento do número de sensores e do número de mensagens que chegam no Drools.

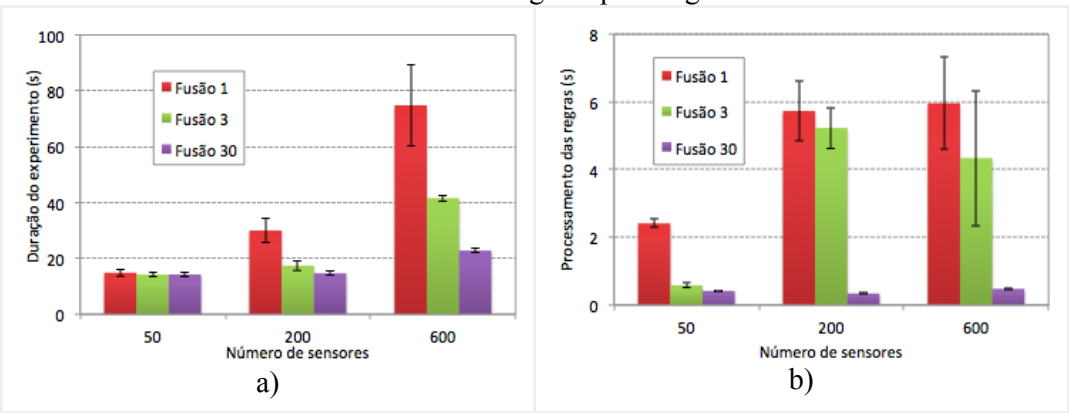


Figura 2.14. Taxa de fusões e tempo de regras do Gerenciador de Contexto

2.5.4. Gerenciamento Dinâmico de NFV

A dinâmica do cenário de computação urbana exige soluções de infraestrutura que acompanham esta característica. A tecnologia NFV apresenta soluções para estas demandas, permitindo a alocação e configuração dos recursos dinamicamente equilibrando qualidade de serviço, custo operacional e investimento.

Um exemplo deste cenário é apresentado em [Heideker e Kamienski 2015] onde o fenômeno *flash crowd*, caracterizado pela grande concentração momentânea de pessoas

em um local público ou privado, é tratado utilizando NFV para prover recursos de infraestrutura de acesso à internet. A Figura 2.15 demonstra a relação entre o uso de uma infraestrutura de telecomunicações utilizando tecnologias tradicionais em ambientes com demanda variável. Neste cenário, quando não há grande demanda de recursos de telecomunicações, há um claro desperdício de recursos como investimento inicial, espaço físico, consumo energético e custo de manutenção, além do consequente impacto ambiental.

Figura 2.15.: Modelo Tradicional de Infraestrutura de Telecomunicações com Demanda Variável.

Sob o ponto de vista da tecnologia NFV, este mesmo cenário poderia utilizar uma infraestrutura dinâmica, onde os recursos são alocados conforme a demanda e dispensados em momentos de ociosidade. A Figura 2.16 demonstra como os recursos necessários para prover acesso à internet neste cenário podem ser particionados em funções de rede virtualizadas.

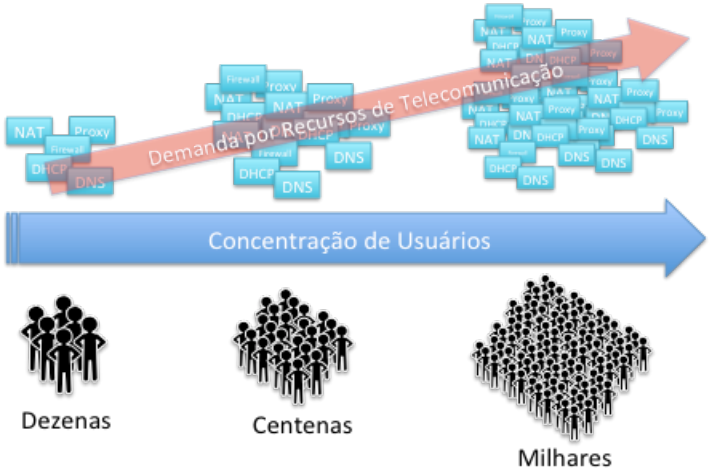


Figura 2.16. Modelo Utilizando NFV Para Fornecer Infraestrutura de Telecomunicações Dinamicamente

O programa WiFi Livre citado anteriormente foi utilizado em um experimento que utiliza a tecnologia NFV para fornecer a infraestrutura de tratamento do tráfego oriundo das praças dinamicamente. O experimento utilizou a estatística de acesso do programa WiFi Livre SP em 117 praças reproduzindo o comportamento dos usuários no ambiente experimental. Neste experimento, o tráfego oriundo das praças é direcionado para um Network Address Translator (NAT) virtualizado. A Figura 2.17 mostra o esquema do experimento, em que o tráfego na VNF NAT é monitorado pelo mecanismo de elasticidade. Quando determinado limiar máximo de uso é atingido, cria uma nova VNF e atua sobre as rotas dos pontos de presença das praças direcionando o tráfego para a nova VNF, balanceando desta forma a carga entre as diversas VNFs disponíveis.

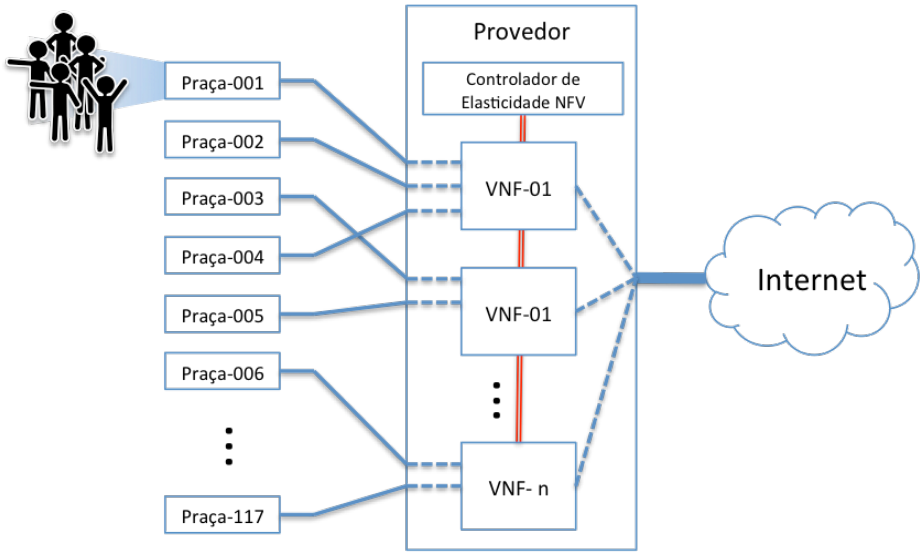


Figura 2.17. Esquema do Experimento de WiFi Gratuito Utilizando Infraestrutura Dinâmica com NFV

Entre as avaliações realizadas deste cenário, a Figura 2.18 mostra a comparação de tempo de download observado pelos usuários das praças utilizando três diferentes tecnologias de virtualização, comparadas a uma abordagem tradicional de tratamento do tráfego com um equipamento dedicado (Direto). Os resultados obtidos mostram que, de maneira geral, existe viabilidade no uso desse mecanismo uma vez que os intervalos de confiança se sobrepõe entre si e os de LXC e XEN se sobrepõe com a configuração direta, ou seja, a abordagem tradicional.

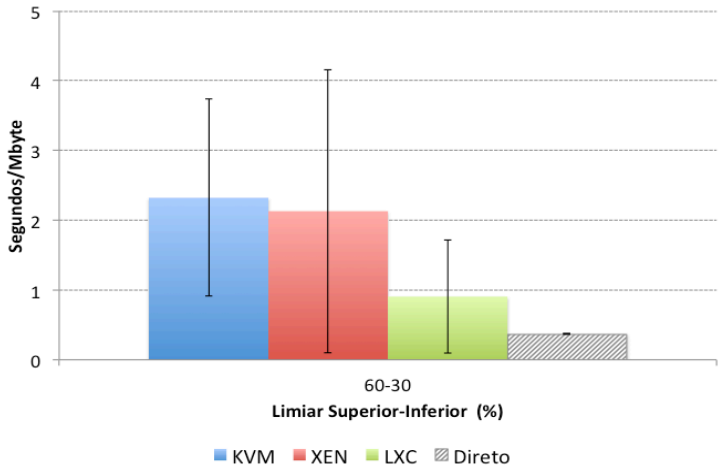


Figura 2.18. Tempo de download observado pelo usuário da Praça em três tecnologias de virtualização e na abordagem com máquina dedicada.

Outra métrica avaliada é o número médio de máquinas virtuais utilizadas no decorrer do experimento, mostrada na Figura 2.19. Esta métrica está relacionada com o custo final da operação de um sistema baseado em NFV, considerando o modelo de negócios utilizado em computação em nuvem, no qual paga-se pelo tempo de uso de uma máquina virtual ou espaço armazenado.

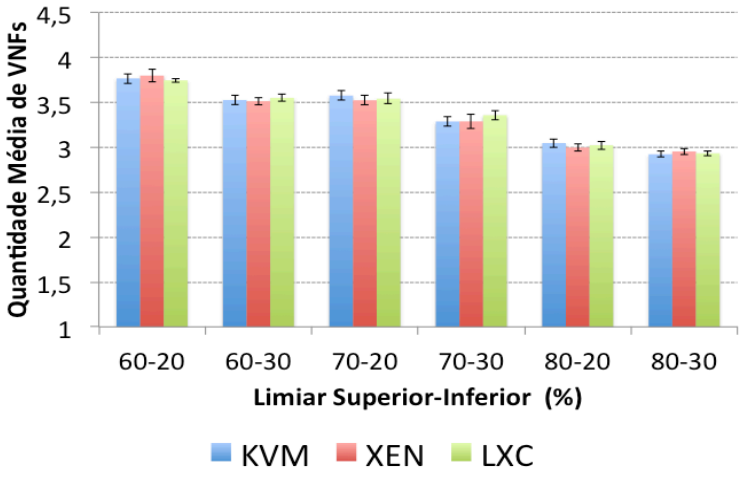


Figura 2.19. Número médio de VNFs utilizadas durante o tempo do experimento

2.6. Discussão: Oportunidades de Pesquisa

Essa seção promove uma discussão sobre as oportunidades de pesquisa na área de redes de computadores e sistemas distribuídos sobre computação urbana e cidades inteligentes.

2.6.1. Segurança e Privacidade do Cidadão

Muitas questões precisam ser resolvidas sobre diferentes aspectos de segurança e privacidade, em diferentes níveis da arquitetura de cidades inteligentes. As pessoas

podem oferecer resistência ao uso de determinados serviços, ao menos que confiem que não oferecem ameaças à privacidade. Os problemas de segurança em tecnologias como IoT, SDN e computação em nuvem ainda não estão resolvidos [Atzori et al 2010] [Kreutz et al 2015]. Por exemplo, a abordagem de SDN permite que novas aplicações de segurança sejam desenvolvidas, mas também traz novos problemas, como ataques aos controladores centralizados Os dispositivos limitados em energia utilizados para sensoriamento pela IoT, precisam de mecanismos de criptografia e autenticação leves, que não consumam muitos recursos. Embora já existam estudos para redes de sensores e RFID, algoritmos e protocolos mais eficientes são necessários.

O gerenciamento de chaves de criptografia é a base de outros mecanismos de segurança e ainda não tem uma solução ideal em redes com limitações de recursos. O gerenciamento de identidades, que consiste de maneiras seguras para atribuir identidades aos objetos e usuários, bem como dos processos de autorização aos recursos, é outro aspecto em aberto. Um middleware que permita a interoperabilidade entre os diversos elementos de maneira segura e ciente de contexto é um elemento importante, para que uma aplicação não divulgue dados pessoais de um usuário e comprometa sua privacidade. Novos métodos para o estabelecimento de confiança devem ser desenvolvidos, como na interação entre objetos e usuários e entre controladores e equipamentos de rede. Para que novos mecanismos de segurança sejam utilizados pelos mais variados tipos de dispositivos de uma rede, é importante a padronização de protocolos. Ainda, políticas e regulamentações que permitam o desenvolvimento de cidades inteligentes precisam ser desenvolvidas e implementadas, já que envolvem questões de segurança de informação nacional e privacidade do cidadão.

2.6.2. Arquiteturas para Sensoriamento de Dados na Internet das Coisas

A heterogeneidade da IoT traz o desafio da interoperabilidade e integração entre os dispositivos que a compõe. A maioria das propostas e padrões para IoT utiliza uma arquitetura orientada a serviços, já largamente estudada e que permite a interoperabilidade entre diversos componentes de uma rede [Zanella *et al.* 2015]. Os serviços fornecidos pela IoT podem utilizar de tecnologias baseadas em serviços, aproveitando a similaridade com outros serviços já utilizados na web, mas agora com objetos inteligentes. Como a IoT é formada tanto por dispositivos com limitações de recursos como dispositivos sem estas limitações, uma arquitetura precisa prever estes dois tipos de entidades. Para cada caso, diferentes tecnologias precisam ser utilizadas. A arquitetura precisa considerar as diferentes tecnologias de formato de dados, de camada de enlace e protocolos de aplicação, transporte e rede. Os dispositivos da IoT também podem ser dos mais diversos: desde sensores e objetos do cotidiano até servidores de alta capacidade, passando por gateways responsáveis pela interconexão dos dispositivos físicos com a infraestrutura principal de rede.

Algumas propostas tem surgido nesta direção, como apresentado em Zanella *et al.* (2015), mas novas propostas e estudos precisam ser feitos. Arquiteturas de referência e plataformas de middleware podem ajudar neste contexto, fornecendo uma camada intermediária entre aplicações e a infraestrutura e padronizando o acesso aos dados. Existem diferentes modelos e plataformas de middleware para IoT, porém na maioria dos casos são incompatíveis entre si e não abordam todos os requisitos desejáveis, como segurança, escalabilidade e ciência de contexto [Razzaque *et al.* 2016] [Delicato *et al.*

2016]. Novas propostas de plataformas que abordem o ambiente IoT utilizando padrões interoperáveis se faz necessária. Arquiteturas de referência que sirvam como base para a construção de novos modelos e plataformas podem auxiliar neste processo.

2.6.3. Modelagem de Dados para Cidades Inteligentes

Os dados utilizados para construir cenários típicos de cidades inteligentes são muito heterogêneos. Podemos ter em um único cenário dados tipicamente armazenados em bancos de dados relacionais e, ao mesmo tempo, dados tipicamente armazenados em diferentes bancos de dados *NoSQL*. Por exemplo, para construir um cenário de mobilidade urbana abrangente pode se considerar os dados dos veículos particulares e públicos (que representam um grande volume de dados em *stream*), o mapa da cidade (que é, na maior parte do tempo, estático e geralmente armazenado em formato de grafo) e dados meteorológicos (que podem ser armazenados em bancos de dados relacionais), entre outras informações. Sendo assim, a modelagem desses dados é um desafio, pois é preciso integrar dados heterogêneos e com formatos diferentes. Em geral, esses dados são armazenados individualmente em bancos de dados típicos para o seu formato, cada um deles possuindo a sua própria modelagem. A integração é feita manualmente por um analista de dados, que fará integrações específicas para os cenários que estão sendo trabalhados. Ou seja, não existe uma modelagem genérica e flexível para integrar dados típicos de uma cidade inteligente.

2.6.4. Balanceamento de dados em Crowdsensing

Os dados gerados através de um método de *crowdsensing* não são uniformemente distribuídos em espaços geográficos e temporais. Em alguns locais, podem ser coletados muito mais dados do que o que realmente é necessário. Um método de subamostragem, por exemplo o método de compressão de sensoriamento, pode ser útil para reduzir a carga de comunicação de um sistema. Ao contrário deste cenário onde os dados são abundantes e deseja-se reduzi-los, também há os casos onde não há dados suficientes para fazer o processamento, ou simplesmente não há nenhum dado para fazer o processamento. Neste caso, há alguns incentivos que podem ser usados para motivar os usuários a contribuir no fornecimento de dados para o processamento. Mas deve-se considerar um orçamento limitado e, então, estudar a melhor configuração para distribuir os incentivos em diferentes locais e períodos de tempos, de modo a maximizar a qualidade dos dados recebidos para uma dada aplicação. Esta área ainda é um desafio de pesquisa a ser explorado.

2.6.5. Distribuição Heterogênea de Dados

Resolver os desafios da computação urbana exige a supervisão de uma ampla gama de fatores, por exemplo, para estudar a poluição do ar é necessário estudar ao mesmo tempo o fluxo de tráfego, a meteorologia, e os usos do solo. Ou seja, os cenários de computação urbana exigem o tratamento de dados heterogêneos. No entanto, as técnicas de aprendizado de máquina e mineração de dados existentes geralmente lidam com um único tipo de dado. Por exemplo, técnicas de visão computacional trabalham com imagens e processamento de linguagem natural com textos.

De acordo com [Zheng *et al.* 2013] tratar igualmente características extraídas de fontes de dados diferentes (por exemplo, simplesmente colocar estas características em um vetor e aplicá-las em um modelo de classificação) não permite alcançar o melhor desempenho. Além disso, utilizar várias fontes de dados em uma única aplicação leva a um espaço de alta dimensão, o que geralmente agrava o problema da dispersão dos dados. Quando não tratadas corretamente, o aumento das fontes de dados pode comprometer o desempenho do modelo ao invés de deixá-lo mais preciso. Por isso, modelos de análise de dados avançados estão chamando a atenção, pois podem aprender com várias fontes de dados heterogêneas, como dados obtidos a partir de sensores, pessoas, veículos e edifícios simultaneamente.

2.6.6. Fusão de Conhecimento

Para aproveitar as informações vindas de uma grande variedade de fontes de dados é necessário fazer uso de tecnologias que efetivamente possam fundir o conhecimento adquirido a partir destas múltiplas fontes de dados heterogêneos. Existem três maneiras principais para atingir este objetivo:

a) Fundir diferentes fontes de dados a nível de características, isto é, tratar diferentes fontes de dados de forma igual e reunir os recursos extraídos de diferentes fontes de dados em um vetor de características. Algum tipo de técnica de normalização deve ser aplicado neste vetor de características antes de usá-lo para alimentar um modelo de análise de dados. Esta é a forma mais comumente vista nas ciências de dados que lidam com fontes heterogêneas.

b) Usar diferentes dados em diferentes fases. Por exemplo, Zheng *et al.* [2011] primeiramente dividiu uma cidade em regiões disjuntas por estradas principais e, em seguida, usou dados de mobilidade humana para reconhecer a configuração problemática da rede de transporte da cidade. Esta é a técnica mais intuitiva quando as pessoas pensam sobre a fusão de dados.

c) Alimentar diferentes conjuntos de dados em diferentes partes de um modelo simultaneamente. Esta técnica baseia-se no conhecimento profundo das fontes de dados e algoritmos. De acordo com os estudos de [Zheng *et al.* 2013], a terceira categoria de método de fusão de dados geralmente tem um desempenho melhor que a primeira, e a segunda categoria pode ser usada simultaneamente com a primeira e a segunda. A primeira e a segunda categorias são bastante intuitivas e têm sido amplamente discutidas na literatura, enquanto que a terceira é apontada como um desafio de pesquisa.

2.7. Conclusão

As cidades estão se tornando mais inteligentes à medida que são introduzidas novas tecnologias nos serviços que diretamente afetam os cidadãos que vivem em grandes aglomerados urbanos. Como o conteúdo desse capítulo mostrou, a inteligência é medida não pela tecnologia em si, mas pelos resultados relativos à melhoria de qualidade de vida da população e eficiência das tarefas executadas pelo poder público. O conjunto de tecnologias usados para criar cidades inteligentes é vasto, mas é possível citar especificamente que a combinação de várias tecnologias como Internet das Coisas, computação em nuvem, softwarização de redes de computadores e Big Data como desempenhando um papel primordial no sentido de avançar a área. Em seu conjunto, o

grupo de tecnologias usadas para viabilizar as cidades inteligentes tem sido chamado de Computação Urbana. É inegável que as cidades continuarão sua evolução amparadas pela tecnologia, independente dos nomes que venham a receber no futuro.

Embora existam muitos desafios a serem vencidos e novos conhecimentos e tecnologias a serem gerados, o ideal de cidades mais inteligentes já pode começar a ser vivenciado com o emprego correto de combinações de tecnologias já existentes. Um aspecto relevante é a necessidade da tecnologia permitir maior participação do cidadão nos processos de governança e interação com os processos de elaboração e implantação de políticas públicas. No futuro, as cidades irão refletir o que a tecnologia em conjunto com a criatividade e os limites impostos pela população permitirem.

Referências

Adams, J. (2009) “The Patologies of Big Data”, Communications of the ACM, vol. 52, no. 8, pp. 36-44.

Al Nuaimi, E., Al Neyadi, H., Mohamed, N., Al-Jaroodi, J. (2015) "Applications of Big Data to Smart Cities", Journal of Internet Services and Applications (JISA), vol. 6, no. 25.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M. (2010) “A View of Cloud Computing”, Communications of the ACM, vol. 53, no. 4, pp. 50-58.

Atzoria, L., Ierab, A., Morabitoc, G. (2010) "The Internet of Things: A survey", Computer Networks, vol. 54, no. 15, pp. 2787-2805.

Baldauf, M., Dustdar, S., Rosenberg, F. (2007) “A survey on context-aware systems”, International Journal of Ad Hoc and Ubiquitous Computing, vol. 2, no. 4, pp. 263-277.

Bejan, A., I., Gibbens, R. J., Evans, D., Beresford, A. R., Bacon, J., Friday, A. (2010) “Statistical Modelling and Analysis of Sparse Bus Probe Data in Urban Areas”, Proceeding of the 13th IEEE International Conference on Intelligent Transportation Systems, pp. 1256-1263.

Bonomi, F. Milito, R., Zhu, J., Addepalli, S., "Fog Computing and its role in the Internet of Things", IEEE Workshop on Mobile Cloud Computing (MCC), pp. 13-16, 2012.

Caragliu, A., Del Bo, C., Nijkamp, P. (2011) “Smart Cities in Europe”, Journal of Urban Technology, vol. 18, no. 2, pp. 65-82.

Cardone, G., Foschini, L., Bellavista, P., Corradi, A., Borcea, C., Talasila, M., Curtmola, R. (2013) "Fostering Participation in Smart Cities: A Geo-Social Crowdsensing Platform", IEEE Communications Magazine, vol. 51, no. 6, pp. 112-119.

Castro-Neto, M., Jeong, Y. S., Jeong, M. K., Han, L. D. (2009) “Online-SVR for short-term traffic prediction under typical and atypical traffic conditions”, Expert Systems with Applications, vol. 36, no. 3, pp. 6164-6173.

Cenedese, A., Zanella, A., Vangelista, L, Zorzi, M.. (2014) “Padova Smart City: An urban Internet of Things experimentation”, IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1-6.

Centenaro, M., Vangelista, L., Zanella, A., Zorzi, M. (2015) “Long-range communications in unlicenses bands: the rising star in the IoT and smart city scenarios”. Disponível em http://arXiv:1510.00620v1.

Cerrudo, C. (2015) “An Emerging US (and World) Threat: Cities Wide Open to Cyber Attacks”, White Paper IOActive.

Cerrudo, C., Hasbini, A., Russell, B. (2015) “Cyber Security Guidelines for Smart City Technology Adoption”, Cloud Security Alliance.

Chandola, V., Banerjee, A., Kumar, V. (2009) "Anomaly detection: a survey", ACM Computing Surveys, vol. 41, no. 3, pp. 1–58.

Chen, M., Mao, S., Liu, Y. (2014) "Big Data: A Survey", Mobile Networks and Applications, vol. 19, no. 2, pp 171-209.

Chen, S. e Zhao, J. (2014) “The requirements, challenges, and technologies for 5G of terrestrial mobile telecommunication”, IEEE Communications Magazine, vol. 52, no. 5, pp. 36–43.

D’Hondt, E. e Stevens, M. (2011) “Participatory noise mapping”, Proceeding of the 9th International Conference on Pervasive Computing, pp. 33-36.

Dean, J. e Ghemawat, S. (2008) “MapReduce: Simplified Data Processing on Large Clusters”, Communications of the ACM, vol. 51, no. 1, pp. 107-113.

Dusparic, I., Harris, C., Marinescu, A., Cahill, V., Clarke, S. (2013) “Multi-agent residential demand response based on load forecasting”, IEEE Conference on Technologies for Sustainability (SusTech), pp. 90-96.

Estellés-Arolas, E., González-Ladrón-de-Guevara, F. (2012), "Towards an integrated crowdsourcing definition", Journal of Information Science, vol. 38, no. 2, pp. 189-200.

ETSI (2012) “Network Functions Virtualization”, ETSI White Paper.

European Comission (2008) “The Future of the Internet”, Report from the National ICT Research Directors Working Group on Future Internet (FI).

Franke, R., Lukowicz, P., Blanke, U. (2015) "Smart crowds in smart cities: real life, city scale deployments of a smartphone based participatory crowd management platform", Journal of Internet Services and Applications (JISA), vol. 6, no. 27, pp. 1-19.

Gandia, R. (2012) “City outlines travel diary plan to determine future transportation needs”. Media, Calgary Sun.

Ge, Y., Xiong, H., Tuzhilin, A., Xiao, K., Gruteser, M., Pazzani, M. (2010) “An energy-efficient mobile recommender system”, Proceedings of 16th SIGKDD Conference on Knowledge Discovery and Data Mining, ACM Press: pp. 899–908.

Hanson, S. e Hanson, P. (1980) “Gender and Urban Activity Patterns in Uppsala, Sweden”, Geographical Review, vol. 70, no. 3, pp. 291-299.

Heideker, A. e Kamienski, C. (2015) “Funções de Rede Virtualizadas em Plataforma de Computação em Nuvem para Cidades Inteligentes”. XIII Workshop em Clouds e Aplicações (WCGA), Vitória-ES, pp. 43–56.

Hendler J. (2014) “Data Integration for Heterogeneous Datasets”, Big Data, vol. 2, no. 4, pp. 205-215.

Herrera, J. C., Work, D., Ban, X., Herring, R., Jacobson, Q., Bayen, A. (2010) "Evaluation of traffic data obtained via GPS-enabled mobile phones: the Mobile Century field experiment", Transportation Research C, vol. 18, pp. 568–583.

Herring R., Hofleitner, A., Abbeel P., Bayen, A. (2010) "Estimating arterial traffic conditions using sparse probe data", Proceeding of the 13th IEEE International Conference on Intelligent Transportation Systems, IEEE Press, pp. 923-929.

Hung, C.C., Chang, C.W., Peng, W.C. (2009) "Mining trajectory profiles for discovering user communities", Proceedings of the 1st ACM SIGSPATIAL GIS Workshop on Location Based Social Networks, ACM Press, pp. 1-8.

Hunter, T., Herring, R., Abbeel, P., Bayen, A. (2009) "Path and travel time inference from GPS probe vehicle data", International Workshop on Analyzing Networks and Learning with Graphs, pp. 1-8.

IBGE (2010), "Sinopse do Senso Demográfico 2010: Brasil", http://www.censo2010.ibge.gov.br/sinopse/index.php?dados=8, acessado em 15/03/2016.

Jiang, S., Ferreira, J., Gonzalez, M. C. (2012) "Discovering urban spatial-temporal structure from human activity patterns", Proceedings of the 1st ACM SIGKDD International Workshop on Urban Computing, ACM Press, pp. 95-102.

Kamienski, C., "Flexible Datacenter Management for Smart Societies", SwitchOn Workshop Miami 2015, Janeiro de 2015.

Kamienski, C., Borelli, F., Biondi, G., Rosa, W., Pinheiro, I., Zyrianoff, I., Sadok, D., Pramudianto, F. (2015) "Context-Aware Energy Efficiency Management for Smart Buildings", IEEE 2nd World Forum on Internet of Things (WF-IoT).

Kanoulas, E., Du, Y., Xia, T., Zhang, D. (2006) "Finding fastest paths on a road network with speed patterns", Proceeding of the 22th International Conference on Data Engineering (ICDE), pp. 1-10.

Karamshuk, D., Noulas, A., Scellato, S., Nicosia, V., Mascolo, C. (2013) "Geo-Spotting: Mining Online Location-based Services for Optimal Retail Store Placement", Proceedings of 19th ACM International Conference on Knowledge Discovery and Data Mining, ACM Press, pp. 793-801.

Kim, G. H., Trimi, S., Chung, J. H. (2014) "Big-data applications in the government sector", Communications of ACM, vol. 57, no. 3, pp. 78–85.

Kreutz, D., Ramos, F. M. V., Verissimo, P. , Rothenberg, C. E., Azodolmolky, S., Uhlig, S. (2015) "Software-Defined Networking: A Comprehensive Survey", Proceedings of the IEEE vol. 103, no. 1, pp. 14-76.

Lathia, N., Froehlich, J., Capra, L. (2010) "Mining Public Transport Usage for Personalised Intelligent Transport Systems", Proceeding of the 10th IEEE International Conference on Data Mining. IEEE Press, pp. 887-892.

Leavitt, N. (2010) "Will NoSQL Databases Live Up to Their Promise?", IEEE Computer, vol. 43, no. 2, pp. 12-14.

Lee, D., Wang, H., Cheu, R., Teo, S. (2004) "Taxi dispatch system based on current demands and real-time traffic conditions", Transportation Research Record: Journal of the Transportation Research Board, vol. 1882, pp. 193–200.

Li, Q., Zheng, Y., Xie X., Chen, Y., Liu, W., Ma, W. (2008) “Mining user similarity based on location history”, 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM Press, pp. 1-10.

Ma H., Zhao D., Yuan P. (2014) "Opportunities in Mobile Crowd Sensing", IEEE Communications Magazine, vol. 52, no. 8, pp 29-35.

Ma, S., Zheng, Y., Wolfson, O. (2013) “T-Share: A Large-Scale Dynamic Taxi Ridesharing Service”, Proceedings of IEEE International Conference on Data Engineering (ICDE), pp. 410-421.

Makris, P., Skoutas, D. N., Skianis, C. (2013) “A Survey on Context-Aware Mobile and Wireless Networking: On Networking and Computing Environments’ Integration”, IEEE Communications Surveys and Tutorials, vol. 15, no. 1, pp. 362-386.

McFedries, P. (2014a) “Urban Computing Reveals the Hidden City”, IEEE Spectrum.

McFedries, P. (2014b) “Urban Computing, Part II: The Language of Smart Cities", IEEE Spectrum.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J. (2008) “OpenFlow: enabling innovation in campus networks” ACM SIGCOMM Computer Communication Review, vol. 38, no. 2: pp. 69-74.

Mijumbi, R., Serrat, J., Gorricho, J . L., Bouten, N., De Turck, F., Boutaba, R. (2015) “Network Function Virtualization: State-of-the-art and Research Challenges”, disponível em: http://arxiv.org/abs/1509.07675.

Monroy-Hernández, A., Farnham, S., Kiciman, E., Counts, S., Choudhury, M. (2013) "Smart Societies: From Citizens as Sensors to Collective Action", ACM Interactions Magazine, vol. 20, no. 4, pp. 16-19.

Naphade, M., Banavar, G., Harrison, C., Paraszczak, J., Morris, R. (2011) "Smarter Cities and Their Innovation Challenges", IEEE Computer, vol. 44, no. 6, pp. 32-39.

Nicolas, M., Stevens, M., Niessen, M. E., Steels, L. (2009) “NoiseTube: Measuring and mapping noise pollution with mobile phones”, Information Technologies in Environmental Engineering, Springer Berlin Heidelberg, pp. 215-228.

Nunes, B., Mendonca, M., Nguyen, X., Obraczka, K., Turletti, T. (2014) "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", IEEE Communications Surveys & Tutorials, 16(3), pp. 1617-1634.

Open Networking Foundation (2012) “Software-Defined Networking: The New Norm for Networks”, ONF White Paper, pp. 1–12.

Open Networking Foundation (2015) "Relationship of SDN and NFV", TR-518, Issue 1.

Open Networking Foundation (2014), "OpenFlow-enabled SDN and Network Functions Virtualization", ONF Solution Brief, February 2014.

Osello A., *et al.* (2013) “Energy saving in existing buildings by an intelligent use of interoperable ICTs”, Energy Efficiency, vol. 6, no. 4, pp. 707-723.

Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D. (2014) "Context Aware Computing for The Internet of Things: A Survey", IEEE Communications Surveys & Tutorials, vol. 16, no. 1.

Pfoser, D. (2008a). Floating Car Data. Encyclopedia of GIS.

Pfoser, D., Brakatsoulas, S., Brosch, P., Umlauft, M., Tryfona, N., Tsironis, G. (2008b). "Dynamic travel time provision for road networks", 16th International Conference on Advances in Geographic Information Systems. ACM Press, pp. 1-4.

Phithakkitnukoon, S., Veloso, M., Bento, C., Biderman, A., Ratti, C. (2010) "Taxi-aware map: Identifying and predicting vacant taxis in the city", Proceeding of the 1st International Joint Conference on Ambient Intelligence, pp. 86-95.

Pires, P. F., Delicato, F. C., Batista, T., Barros, T., Cavalcante, E., Pitanga, M. (2015) "Plataformas para a Internet das Coisas", Minicursos SBRC - Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos.

Postscapes. (2015) "Anatomy of a Smart City", Postscapes Infographic, http://postscapes.com/anatomy-of-a-smart-city-full, acessado em 15/03/2016.

Rana, R. K., Chou, C. T., Kanhere, S. S., Bulusu, N., Hu, W. (2010) "Ear-phone: an end-to-end participatory urban noise mapping system", Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks, pp. 105-116.

Ratti, C., Sobolevsky, S., Calabrese, F., Andris, C., Reades, J., Martino, M., Claxton, R., Strogatz, S. H. (2010) "Redrawing the Map of Great Britain from a Network of Human Interactions", PLoS ONE, vol. 5, no. 12.

Ratusznei, J., Barbosa, W., Pinheiro, N., Melo, R., Kamienski, C. (2015) "Uma Rede WiFi Aberta de Larga Escala como Infraestrutura para Cidades Inteligentes", Seminário Integrado de Software e Hardware (SEMISH).

Razzaque, M. A., Milojevic-Jevric, M., Palade, A.. Clarke, S. (2016) "Middleware for Internet of Things: A Survey", IEEE Internet of Things Journal, vol. 3, no. 1, pp. 70-95.

Rosa, R., Siqueira, M, Rothenberg, C. E. , Barea, E., Marcondes, C. (2014) "Network Function Virtualization: Perspectivas, Realidades e Desafios", Minicursos SBRC - Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos.

Sakr, S., Liu, A., Batista, D. M., Alomari, M. (2011) "A Survey of Large Scale Data Management Approaches in Cloud Environments", IEEE Communications Surveys & Tutorials, vol. 13, no. 3, pp. 311-326.

Sanchez, L., Muñoz, L., Galache, J. A., Sotres, P., Santana, J. R., Gutierrez, V., Pfisterer, D. (2014) "SmartSantander: IoT experimentation over a smart city testbed" Computer Networks, vol. 61, pp. 217-238.

Schaffers, H., Komninos, N., Pallot, M., Trousse, B., Nilsson, M., Oliveira, A. (2011) "Smart cities and the future internet: towards cooperation frameworks for open innovation", The Future Internet, Springer, pp. 431-446.

Shaheen, S., Guzman, S., Zhang, H. (2010) "Bikesharing in Europe, the Americas, and Asia: past, present, and future", Transportation Research Record: Journal of the Transportation Research Board, vol. 2143, pp. 159-167.

Silva, T., Loureiro, A. (2015) "Computação Urbana: Técnicas para o Estudo de Sociedades com Redes de Sensoriamento Participativo", 34º Jornadas de Atualização em Informática (JAI), Congresso da SBC.

Simões, R., Kamienski, C. (2014) "Elasticity Management in Private and Hybrid Clouds", 7th IEEE International Conference on Cloud Computing (Cloud 2014).

Thiagarajan A., Ravindranath, L., Lacurts, K., Madden, S., Balakrishnan, H., Toledo, S., Eriksson, J. (2009) “VTrack: accurate, energy-aware road traffic delay estimation using mobile phones”, 7th ACM Conference on Embedded Networked Sensor Systems, pp. 85-98.

Thusoo, A. (2010) “Data Warehousing and Analytics Infrastructure at Facebook”, ACM SIGMOD, pp. 1013-1020.

Toch, E. (2014) "Crowdsourcing privacy preferences in context-aware applications”, Personal and Ubiquitous Computing, vol. 18, no. 1, pp. 129-141.

Tonguz, O. K., Wisitpongphan, N., Parikh, J. S., Bai, F., Mudalige, P., Sadekar, V. K. (2006) “On the Broadcast Storm Problem in Ad hoc Wireless Networks.”, 3rd International Conference on Broadband Communications, Networks and Systems (BROADNETS), pp. 1–11.

Watkins, K., Ferris, B., Borning, A., Rutherford, S., Layton, D. (2011) “Where Is My Bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders”, Transportation Research Part A vol. 45, pp. 839–848.

Yamamoto, K., Uesugi, K., Watanabe, T. (2010) “Adaptive routing of cruising taxis by mutual exchange of pathways”, Knowledge-Based Intelligent Information and Engineering Systems, vol. 5178, pp. 559–566.

Yuan, J., Zheng, Y., Xie, X. (2012) “Discovering regions of different functions in a city using human mobility and POIs”, Proceedings of 18th SIGKDD Conference on Knowledge Discovery and Data Mining. ACM Press, pp. 186-194.

Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M. (2014) “Internet of Things for Smart Cities”, IEEE Internet of Things Journal, vol. 1, no. 1, pp. 22-32.

Zhang, F., Wilkie, D., Zheng, Y., Xie, X. (2013) “Sensing the Pulse of Urban Refueling Behavior”, Proceedings of the 15th International Conference on Ubiquitous Computing, ACM Press, pp. 13-22.

Zheng, Y., Zhang, L., Xie, X., Ma, W. Y. (2009) “Mining interesting locations and travel sequences from GPS trajectories”, Proceedings of the 18th International Conference on World Wide Web, ACM Press, pp. 791-800.

Zheng, Y. (2011) “Location-Based Social Networks: Users”. In: Computing with spatial trajectories. Eds. Zheng, Y. e Zhou, X. Springer, pp. 243-276.

Zheng, Y. (2012) “Tutorial on Location-Based Social Networks”, Proceedings of 21st International Conference on World Wide Web, pp. 1-2.

Zheng, Y. e Xie, X. (2010) “GeoLife: A Collaborative Social Networking Service among User, location and trajectory”, IEEE Data Engineering Bulletin, vol. 33, no. 2, pp. 32-40.

Zheng, Y., Liu, Y., Yuan, J. Xie, X. (2011) “Urban Computing with Taxicabs”, Proceedings of the 13th International Conference on Ubiquitous Computing, ACM Press, pp. 89-98.

Zheng, Y., Capra, L., Wolfson, O., Yang, H. (2014) "Urban Computing: Concepts, Methodologies, and Applications", ACM Transactions on Intelligent Systems and Technology, vol. 6, no. 2, article 38.

Capítulo 3

Engenharia de Tráfego em Redes Definidas por Software

Jeandro de M. Bezerra[1], Antônio Janael Pinheiro[1], Michel S. Bonfim[2], José A. Suruagy Monteiro[1] e Divanilson R. Campelo[1]

[1]Universidade Federal de Pernambuco - Centro de Informática (CIn)
[2]Universidade Federal do Ceará - Campus de Quixadá

Abstract

Currently, applications and services present in data centers have generated a significant increase in network traffic. This property has given rise to problems in both intra and inter data center scenarios due to the dynamic characteristic and unpredictability of the generated traffic. The consolidation of Software Defined Networking (SDN) technologies have enabled the implementation of use cases that make possible to improve the network management through Traffic Engineering (TE) techniques. In this short course, attendees will have a chance to know the main challenges and the state of the art of TE solutions implemented with the help of the SDN paradigm, along with two practical case studies.

Resumo

Atualmente, aplicações e serviços presentes em data centers têm gerado um aumento significativo no tráfego na rede. Esta propriedade tem gerado problemas tanto nos cenários intra como inter data center devido à característica dinâmica e imprevisibilidade do tráfego gerado. A consolidação de tecnologias de Redes Definidas por Software (SDN) vêm possibilitando a implementação de casos de uso que permitem melhorar o gerenciamento da rede através de técnicas de Engenharia de Tráfego (ET). Neste minicurso, os participantes terão a oportunidade de conhecer os desafios e o estado da arte das soluções de ET implementadadas com o auxílio do paradigma SDN, bem como dois estudos de caso práticos.

Este trabalho utilizou recursos da FUNCAP, FACEPE e CNPq.

3.1. Introdução

Atualmente, os serviços de computação em nuvem presentes principalmente nos grandes *data centers* são responsáveis por uma parcela significativa do aumento do tráfego de dados na Internet. Por causa da característica dinâmica e da imprevisibilidade do tráfego gerado por estes serviços, este acréscimo impõe desafios nos cenários intra e inter *data center*, levando à necessidade de novas arquiteturas e mecanismos de Engenharia de Tráfego (ET) mais eficientes que os tradicionais implementados, por exemplo, para os protocolos BGP, OSPF e MPLS.

ET em WAN (*Wide Area Network*) significa que a comunicação entre pares ocorre por diferentes caminhos na rede, cada um com gargalos diferentes. Cada serviço possui requisitos diferentes de largura de banda, latência e perda [Kumar et al. 2015]. Em redes intra *data center*, os diversos tipos de aplicações de tempo real, dentre as quais destacam-se migração de máquinas virtuais, vídeo e *big data analytics*, necessitam de baixa latência. Sendo assim, são necessários esquemas de balanceamento de carga para garantir a eficiência operacional da rede. Os esquemas tradicionais de balanceamento de carga baseados em *hashing* de fluxos, como, por exemplo, ECMP (*Equal Cost Multi-path*), causam congestionamento quando ocorrem colisões de *hash* e não atuam com bom desempenho em topologias com assimetria [He et al. 2015, Al-Fares et al. 2010]. Nos cenários com WANs, há um aprovisionamento de enlaces além do necessário em torno de 30–40 % da utilização média para, eventualmente, ocultar possíveis falhas nos roteadores para os clientes. Esse fator leva a um uso ineficiente do enlace [Jain et al. 2013]. Os desafios para a construção de WANs para interconexão de vários *data centers*, denominadas de inter-DC WANs, envolvem, dentre outros pontos, custo elevado dos enlaces, pois geralmente estes enlaces são dedicados e de alta capacidade. Além disso, os *switches* possuem *hardware* limitado para criar várias regras de encaminhamento e pouca flexibilidade devido à complexidade para aplicar imparcialidade na alocação de recursos entre serviços que competem entre si [Hong et al. 2013].

A disseminação e consolidação das tecnologias de Redes Definidas por Software (*Software Defined Networking - SDN*) vêm possibilitando a implementação de casos de uso que possibilitam melhorar o gerenciamento da rede através de técnicas de Engenharia de Tráfego (ET) centralizadas no controlador. Estas novas arquiteturas têm modificado a forma de como as redes são projetadas e gerenciadas [Moshref et al. 2014] por meio do desacoplamento do plano de controle e do plano de dados, permitindo, assim, uma maior flexibilidade e possibilidade de adicionar novas funcionalidades à rede. Além disso, novas aplicações SDN estão sendo desenvolvidas e integradas à nuvem [Barros et al. 2015]. *Data centers* são infraestruturas fundamentais para o desenvolvimento de aplicações que se beneficiam das características dos recursos físicos e de paradigmas da computação em nuvem para auxiliar no seu processamento. Atualmente, grandes empresas estão adotando SDN em seus *data centers* devido à facilidade de gerenciamento e melhoria de desempenho da rede. Outro viés é a possibilidade de interação mútua entre *big data* e SDN, possibilitando o desenvolvimento e execução de ET de forma massiva. Os motivos são abordados em [Cui et al. 2016]:

1. Facilidade de obtenção de *big data* sobre informações de tráfego e de falhas.

2. Quaisquer formatos e granularidade dos fluxos de tráfego podem ser utilizados para ET.

3. Facilidade de aplicar políticas de ET nos *switches* na rede do *data center* através da modificação das tabelas de fluxo dentro dos *switches*.

As perspectivas e desafios de pesquisa com SDN e ET têm recebido destaque entre pesquisadores. Como exemplo, o trabalho de [Batista et al. 2015] apresenta entrevistas com cinco pesquisadores de referência da comunidade de redes. Uma das perguntas questionava que tema deveria ser colocado para alunos ingressantes em programas de doutorado. Raouf Boutaba respondeu com a seguinte pergunta: *"how to develop algorithms that leverage SDN abstractions to make traffic engineering decisions and ultimately better utilize the network resources?"*. Pela hipótese levantada, percebe-se o desafio levantado e o bom campo para desenvolvimento de pesquisas.

Este capítulo tem alguma interseção com outros capítulos já publicados em edições passadas de minicursos. No minicurso [Verdi et al. 2010], os autores apresentam as técnicas de ET utilizadas em *data centers* que têm sido exploradas e contribuições de melhoria têm sido realizadas através da utilização de SDN. O trabalho de [Barros et al. 2015] aborda a integração entre redes de *data center* com nuvem e SDN, porém não são considerados aspectos de ET em cenários de *data center*. O trabalho de [Costa et al. 2012] aborda o impacto de grandes massas de dados em redes de computadores e em *data centers*, bem como tecnologias e mecanismos utilizados para melhorar o gerenciamento de grandes massas de dados em ambientes de *data center*.

De uma forma geral, o objetivo principal deste minicurso é fornecer ao público interessado os principais conceitos que possibilitem a utilização e construção de ambientes para experimentação e desenvolvimento de técnicas de ET em SDN. O minicurso apresenta tanto um enfoque teórico como prático, com informações importantes e dicas úteis baseadas em experimentos práticos. O capítulo está organizado da seguinte forma. A Seção 3.2 aborda os princípios de ET e sua evolução nas redes de computadores. Na Seção 3.3, são discutidos paradigmas de SDN e protocolos utilizados em soluções de ET. A Seção 3.4 elenca os principais trabalhos de ET com SDN implementados em ambientes intra e inter *data center*, seguida pela Seção 3.5, que aborda dois estudos de caso. Por fim, a Seção 3.6 apresenta as considerações finais sobre as soluções de ET em SDN e tece alguns comentários a respeito das dificuldades de implementação e tendências futuras para o uso de ET.

3.2. Engenharia de Tráfego

A Engenharia de Tráfego é um atributo fundamental para todos os tipos de redes. Em geral, os principais objetivos de ET são efetuar balanceamento de carga, maximizar a utilização da rede e minimizar o consumo de energia [Agarwal et al. 2013]. Nesta seção, são apresentados os principais aspectos da evolução de técnicas de ET nas tecnologias descritas nas subseções a seguir. São também elencados os mecanismos de ET implementados nas tecnologias de redes indicadas na Figura 3.1.

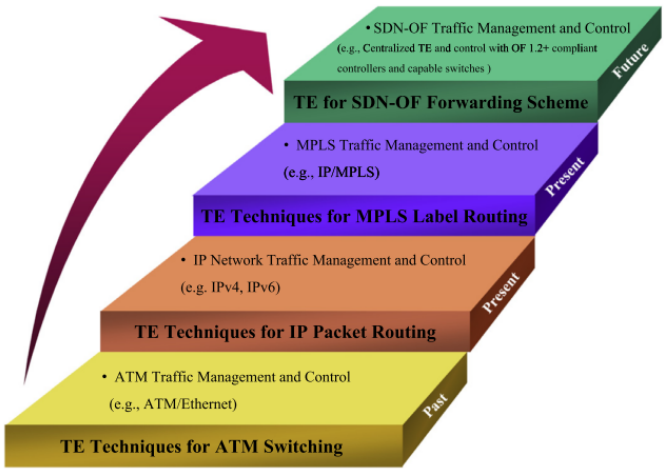


Figura 3.1. Evolução da ET. Extraída de [Akyildiz et al. 2014]

3.2.1. Engenharia de Tráfego em Redes ATM

A tecnologia ATM (*Asyncronous Transfer Mode*), desenvolvida no final da década de 80, foi planejada e desenvolvida com mecanismos de QoS (*Quality of Service*) nativos. ATM utiliza comutação por célula de tamanho fixo, com 53 octetos (48 de informação e 5 de cabeçalho), possuindo a capacidade de transportar diversos tipos de serviços simultaneamente através dessas células.

Os requisitos de QoS incluem vazão, atraso e variação estatística do atraso (*jitter*). Vazão alta e baixa latência são fatores críticos para o controle de congestionamento, principalmente em serviços de multimídia. ATM adota o esquema de controle de congestionamento preventivo para tráfego CBR (*Constant Bit Rate*) e VBR (*Variable Bit Rate*) [Akyildiz et al. 2014], no qual o nó origem não espera que o congestionamento ocorra. Os nós atuam de forma preventiva através do controle dos fluxos de tráfego nos pontos de entrada da rede. Isso funciona em redes ATM, pois há orientação à conexão e a decisão para novos fluxos na rede é tomada com base no conhecimento do estado da rota que o tráfego deve seguir. Esse controle preventivo pode ser executado de três formas: controle de admissão, policiamento da banda e classificação de tráfego [Akyildiz et al. 2014].

3.2.2. Engenharia de Tráfego em Redes IP

A ET em redes IP é concentrada no roteamento. Encontrar rotas utilizando o algoritmo de menor caminho (*Shortest Path First* — SPF) através da definição de pesos nos enlaces por meio dos IGPs (*Interior Gateway Protocols*), tais como o OSPF (*Open Shortest Path First*) ou IS-IS (*Intermediate System–Intermediate System*), foram os primeiros mecanismos implementados em redes IP [Wang et al. 2008]. Roteadores utilizam esses protocolos para atualização dos custos dos enlaces e para a construção de uma visão completa da topologia da rede dentro de Sistemas Autônomos (SA) [Akyildiz et al. 2014]. Esses protocolos podem fazer com que o tráfego siga pelos mesmos caminhos, o que pode levar a um congestionamento na rede. Além disso, mudanças nos pesos dos enlaces

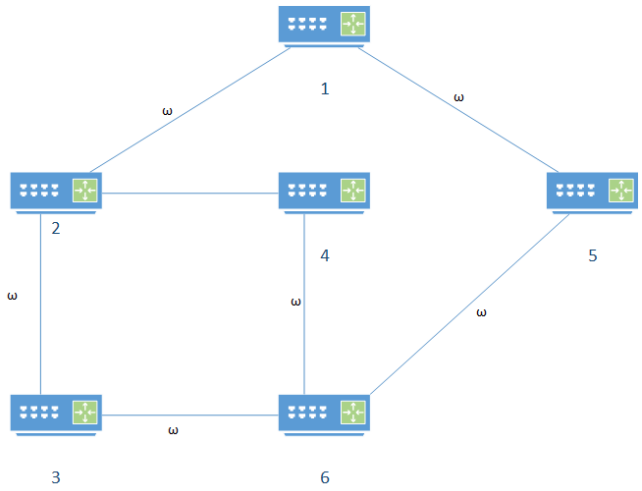


Figura 3.2. Exemplo de ECMP

estáticos afetam o roteamento de todo o conjunto de fluxos.

O ECMP[1] (*Equal-Cost Multi-Path*) é um mecanismo utilizado para ET em redes IP intra-domínio. Se existirem múltiplos caminhos com o mesmo custo para alcançar o mesmo destino, o tráfego é dividido igualmente entre eles. O exemplo da Figura 3.2, extraído de [Medhi 2010], ilustra o funcionamento do ECMP. Os caminhos entre os roteadores 1 e 6 têm custos iguais atribuídos por ω. Então, o tráfego do roteador 1 para o roteador 6 será igualmente dividido no roteador 1 ao longo de dois enlaces direcionais, $1 \rightarrow 2$ e $1 \rightarrow 5$. O tráfego do roteador 1 que chegou ao roteador 2 será igualmente dividido pelos seus dois enlaces de saída, $2 \rightarrow 3$ e $2 \rightarrow 4$. Portanto, do tráfego do roteador 1 destinado ao roteador 6, 25% chegará ao roteador 6 através dos enlaces $3 \rightarrow 6$, 25% através do enlace $4 \rightarrow 6$ e os 50% restantes através do enlace $5 \rightarrow 6$

Alguns trabalhos, tais como [Greenberg et al. 2005, Caesar et al. 2005], inovaram na questão do roteamento em redes. Greenberg et al, (2005) propuseram uma arquitetura, denominada de 4D, que, através da separação dos planos de controle e de dados, separa completamente a decisão de roteamento lógico dos protocolos que controlam as interações entre elementos da rede. Em [Caesar et al. 2005], os autores implementaram a RCP (*Routing Control Platform (RCP)*), uma plataforma logicamente centralizada que separa o plano de encaminhamento para prover escalabilidade e evitar problemas de complexidade em arquiteturas iBGP (*internal Border Gateway Protocol*). Essas arquiteturas serviram de base para que pesquisadores em SDN desenvolvessem a separação dos planos de dados e de controle e, posteriormente, implementassem controladores SDN e *switches* Openflow [Akyildiz et al. 2014].

3.2.3. Engenharia de Tráfego em Redes MPLS

O conceito de ET foi inicialmente introduzido em ambientes MPLS (*Multi-Protocol Label Switching*). O MPLS foi desenvolvido para suprir problemas de ET em redes IP e

[1]Definido na RFC 2992 - *Analysis of an Equal-Cost Multi-Path Algorithm*

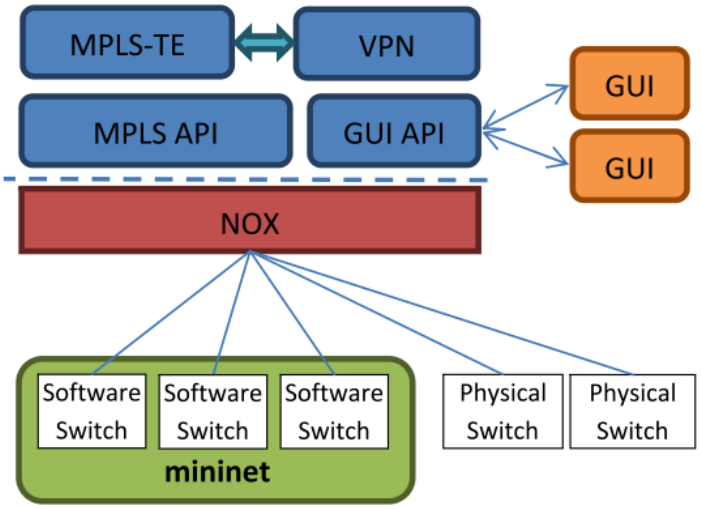


Figura 3.3. Arquitetura da Rede física e controlador. Extraída de [Sharafat et al. 2011]

é amplamente utilizado em provedores de serviço [Wang et al. 2008].

A grande vantagem da ET em redes MPLS é a sua capacidade de realizar, de forma eficiente, roteamento explícito entre origem e destino através da adoção de LSPs (*Label Switched Paths*). Para o estabelecimento das rotas, é utilizado um protocolo de distribuição de rótulos denominado de (*Label Distribution Protocol* - LDP). Nessa fase, ocorre troca de rótulos entre LSRs (*Label Switch Routers*). Atualmente, o RSVP-TE (*Resource ReSerVation Protocol with Traffic Engineering extension*) é o protocolo utilizado para distribuição de rótulos [Medhi 2010].

O MPLS-TE (*Multi-Protocol Label Switching Traffic Engineering*) pode ser combinado com o *OpenFlow* para o desenvolvimento de soluções que reduzam a complexidade do plano de controle e proporcionem simplicidade ao gerenciamento da rede [Sharafat et al. 2011]. O plano de controle de soluções baseadas apenas em MPLS-TE é complexo, pois há a necessidade de se utilizarem vários protocolos, como, por exemplo, RSVP-TE, iBGP e LDP. Este fato torna complexa a implementação de novas aplicações e serviços e aumenta o tempo de processamento na rede. Sharafat et. al, (2011) demonstraram a integração do MPLS-TE e MPLS-VPN com o *OpenFlow* utilizando o controlador NOX, conforme ilustrado na Figura 3.3.

Os autores comprovaram a eficiência da solução[2] através da configuração de um *testbed* emulando uma WAN com múltiplas instâncias do OpenvSwitch. Esses *switches* são conectados à rede dos *switches* físicos, ambos com suporte a *OpenFlow* e MPLS especificado no *OpenFlow* 1.1. Os *switches* foram modificados para que fosse possível operar com o plano de dados do MPLS.

[2]A demonstração pode ser acessada em https://youtu.be/EpttFVKUrzs

3.3. Redes Definidas Por Software: paradigma e protocolos

Esta seção discorre sobre o paradigma de Redes Definidas por *Software* (subseção 3.3.1 e descreve dois dos protocolos utilizados como *API Southbound* em soluções SDN de Engenharia de Tráfego: o *OpenFlow* (subseção 3.3.2) e o LISP (subseção 3.3.3).

3.3.1. Redes Definidas por Software

Redes IP tradicionais são complexas e difíceis de gerenciar. Para configurar políticas de rede, os operadores de rede precisam configurar cada dispositivo, individualmente, usando comandos de baixo nível e específicos por fabricante. Além disso, mecanismos de reconfiguração automática e resposta, necessárias para adaptação da rede a falhas e mudanças de carga, são praticamente inexistentes nas redes atuais. Tais características reduzem a flexibilidade da rede, bem como dificulta a sua evolução e inovação, tornando-as "engessadas" [Kreutz et al. 2015].

Redes Definidas por *Software* (*Software Defined Networking* - SDN) são um novo paradigma de redes que surgiu inicialmente da dificuldade de os pesquisadores desenvolverem e testarem novas soluções e protocolos em ambientes de produção. Esta dificuldade ocorre porque o código fonte executado em *switches* e roteadores comerciais é fechado [McKeown et al. 2008]. A principal característica das SDNs é a separação dos planos de controle e de dados da rede, antes verticalmente integrados nos elementos de rede.

O plano de controle, representado por uma entidade chamada de controlador, é responsável pelas decisões de como tratar o tráfego na rede. O controlador pode ser executado em servidores comerciais, separados dos equipamentos da rede. Por sua vez, o plano de dados, presente nos dispositivos de rede, é responsável pelo encaminhamento dos dados de acordo com um conjunto de regras estabelecidas. Tais regras são criadas e gerenciadas pelo controlador, que exerce um controle direto sobre o estado dos elementos do plano de dados através de uma API (*Application Interface Programming*) [Kreutz et al. 2015].

Na figura 3.4, estão ilustrados os três planos de uma arquitetura SDN e as APIs responsáveis pelas interações entre elas. A API *Northbound* é responsável pela comunicação entre o plano de aplicação e o plano de controle, fornecendo suporte a aplicações de engenharia de tráfego, roteamento, *firewall*, QoS, etc. A API *Southbound* é responsável pela comunicação do plano de dados com o plano de controle. O protocolo *OpenFlow* é uma das APIs *Southbound* mais utilizadas.

A separação entre os planos de dados e de controle simplifica a aplicação de políticas de rede e facilita a sua configuração e evolução. Ela é chave para a flexibilidade da rede, quebra o problema do controle da rede em partes tratáveis e torna mais fácil o gerenciamento e a introdução de novas aplicações, como Engenharia de Tráfego.

3.3.2. O Protocolo *OpenFlow*

O protocolo *OpenFlow* é um dos principais responsáveis pela implementação atual de SDN aberta e baseada em padrões. O desenvolvimento do *OpenFlow* teve início em 2007 e sua evolução tem recebido contribuições da academia e indústria. Concebido originalmente por pesquisadores da Universidade de Stanford e da Universidade da Califórnia

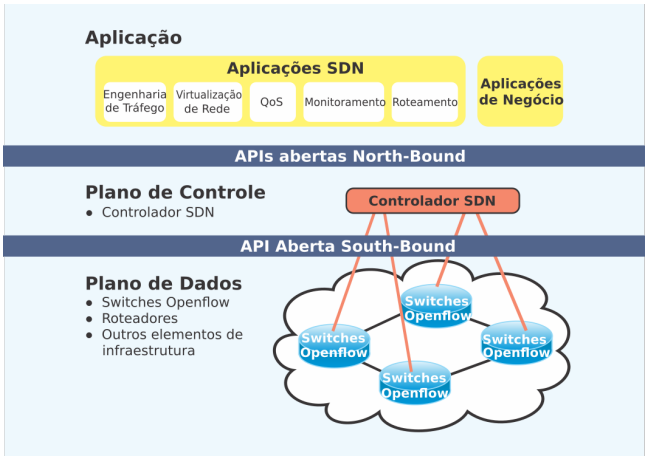


Figura 3.4. Arquitetura de uma SDN. Adaptada de: [Akyildiz et al. 2014]

em Berkeley, o padrão tem sido mantido pela *Open Networking Foundation* (ONF[3]). A ONF é uma organização direcionada ao usuário, dedicada à promoção e adoção de SDN através do desenvolvimento de padrões abertos e que possui como membros algumas das principais empresas da área de tecnologia, tais como Google, Microsoft, Cisco, AT&T, Juniper, Oracle, dentre outras.

A Figura 3.5 ilustra os principais componentes de uma rede SDN *OpenFlow*. Os ***switches OpenFlow*** são responsáveis pelo encaminhamento dos pacotes de dados na rede (isto é, plano de dados). Eles formam a infraestrutura que envia os pacotes de acordo com as regras criadas e mantidas pelo controlador. Tais regras são armazenadas em uma ou mais **Tabelas de Fluxo** (*Flow Tables*) e/ou uma **Tabela de Grupos** (*Group Table*), que são utilizadas no encaminhamento dos pacotes.

Por outro lado, o **controlador *OpenFlow*** é o principal componente deste tipo de rede. Em termos arquiteturais, o controlador dá suporte às aplicações de rede, determinando as regras a serem armazenadas e aplicadas pelos *switches*. Existem vários controladores *OpenFlow* no mercado, dentre os quais podem ser citados: NOX[4], POX[5], ONOS[6], Floodlight[7], OpenDayLight [ODL 2016] e Ryu[8].

O controlador usa um **canal seguro e criptografado** para, através do protocolo *OpenFlow*, realizar o gerenciamento dos *switches*. Se um pacote recebido em um *switch OpenFlow* tem correspondência com uma regra que inclui uma ação de redirecionamento para um controlador *OpenFlow*, este *switch* encapsulará o pacote em uma mensagem ***Packet-in*** e a encaminhará ao controlador. Esta mensagem pode conter o pacote total ou apenas parte de seu cabeçalho, mas, neste último caso, o *switch* precisa armazenar o pacote completo em um *buffer*.

[3]https://www.opennetworking.org/index.php
[4]https://github.com/noxrepo/nox
[5]https://github.com/noxrepo/pox
[6]http://onosproject.org/
[7]http://www.projectfloodlight.org/
[8]https://osrg.github.io/ryu/

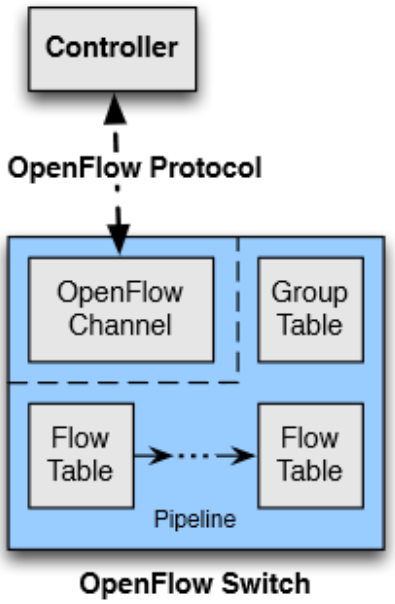


Figura 3.5. Componentes *OpenFlow*. Fonte: [ONF 2015]

Após processada a mensagem *Packet-in*, o controlador, em resposta ao *switch*, deve enviar uma mensagem ***Packet-out*** para especificar qual é a porta de saída que o equipamento cliente (*switch*) deve utilizar para encaminhar o pacote (anteriormente encapsulado por uma mensagem *Packet-in*). O *Packet-out* pode conter o pacote completo ou uma referência a um pacote guardado no *buffer* do cliente.

Paralelamente ao *Packet-out*, o controlador pode enviar uma mensagem ***Modify-state (Flow-Mod)***, cujo principal propósito é adicionar, apagar e modificar entradas de fluxo nas tabelas. O objetivo é evitar que pacotes similares sejam processados novamente pelo controlador.

Desde a versão 1.1, o *OpenFlow* passou a dar suporte a MPLS na definição das suas regras de fluxo, permitindo, assim, o uso desta tecnologia na implementação de soluções de ET [Kreutz et al. 2015]. Neste sentido, surgiram trabalhos que se concentram no uso desta funcionalidade para prover soluções de ET [Sharafat et al. 2011, Gharbaoui et al. 2014]. Em ambientes de redes de *data center* (DCN, *Data Center Network*), o *OpenFlow* possibilita maior flexibilidade no gerenciamento de fluxos nos *switches* para implementar os mecanismos de ET (por exemplo, balanceamento de carga, monitoramento de tráfego, etc.). Dentre as principais vantagens de se utilizar *OpenFlow* para ET, destacam-se:

- Facilidade na obtenção de estatísticas de tráfego e informações de falhas em dispositivos da rede. Essas informações são essenciais para ET na tomada de decisões.

- O *OpenFlow* pode reduzir o atraso no processamento do tráfego, pois todo o tráfego é processado através de fluxos e não por pacotes. Isso permite a utilização de níveis

distintos de granularidade de agregação, como, por exemplo, agregar fluxos entre pares de *switches* ToR, fluxos entre dois *hosts*, etc.

3.3.3. O Protocolo LISP

Desde o seu surgimento, o protocolo LISP ***Locator/Identifier Separation Protocol*** tem recebido bastante atenção da academia e da indústria, sobretudo devido as possibilidades de Engenharia de Tráfego que permite. Como resultado da padronização e esforços de pesquisa, o LISP tem sido aplicado em cenários que vão além da sua proposta original, como ambientes SDN.

O LISP foi inicialmente proposto pela Cisco como uma solução para o problema de escalabilidade da Internet. Este protocolo utiliza dois espaços de endereçamento: um espaço para a identificação e outro para a localização. O primeiro é denominado ***Endpoint Identifiers*** **(EID)** e é utilizado para a identificação dos *hosts*. O segundo é associado aos dispositivos que compõem o sistema global de roteamento, e é conhecido como ***Routing Locators*** **(RLOCs)**. Os RLOCs são utilizados no roteamento de pacotes, permitindo a localização dos *hosts* [Farinacci et al. 2015]. Ambos EIDs e RLOCs são geralmente endereços IPv6 ou IPv4.

Além disso, o LISP também define um mecanismo de encapsulamento para pacotes endereçados com EIDs na transmissão que utiliza RLOCs para roteamento e encaminhamento [Farinacci et al. 2015], criando o chamado **Túnel LISP**. Além disso, ele também define um sistema de mapeamento entre os dois espaços de endereçamento que, juntamente com o mecanismo de encapsulamento, permite que o tráfego, originado por dispositivos que usam EIDs não-roteáveis, seja transportado através da infraestrutura da rede, a qual roteia os pacotes usando RLOCs.

Uma rede LISP composta por um conjunto de EIDs e RLOCs e com suas próprias políticas de mapeamento é conhecida como **Site LISP**. Um *site* LISP pode se comunicar com outros *sites* LISP e/ou com *sites* não-LISP. Para tanto, são necessários elementos para disponibilizar tal infraestrutura de comunicação: os roteadores LISP, o sistema de mapeamento LISP e o protocolo LISP [Farinacci et al. 2015]. Os roteadores LISP podem ser dos seguintes tipos:

- ***Ingress Tunnel Router (ITR):*** geralmente reside em um *site* LISP. Um ITR recebe e encapsula (utilizando um endereço RLOC) os pacotes originados de fontes internas (dentro do *site* LISP com um EID específico) e destinados para fora do *site*;
- ***Egress Tunnel Router (ETR):*** geralmente reside em um *site* LISP. Um ETR recebe um pacote IP encapsulado (RLOC) com o protocolo LISP e envia o pacote IP desencapsulado (EID) para um *host* específico;
- ***Proxy-ITR (PITR):*** o PITR funciona como um ITR, mas age em nome de um *site* não-LISP que deseja enviar pacotes com destino para um *site* LISP;
- ***Proxy-ETR (PETR):*** o PETR fuciona como um ETR, mas age em nome de um *site* LISP que deseja enviar pacotes destinados ao um *site* não-LISP;

- ***Reencapsulating Tunnel Router (RTR):*** o RTR é um roteador que age como um ETR ou PETR, desencapsulando pacotes cujo endereço de destino do cabeçalho IP mais externo contém um dos seus RLOCs. Em seguida, ele age como um ITR ou PITR, tomando a decisão de como encapsular o pacote.

Por outro lado, o sistema de mapeamento LISP é responsável pelo registro de mapeamentos EID-RLOC e pelo processamento de suas buscas, sendo composto pelos seguintes componentes:

- ***Map-Server:*** componente responsável por receber e processar as mensagens de registro dos ETRs (mensagens *Map-Register*) com mapeamentos EID-RLOC, que posteriormente serão armazenadas em um banco de dados distribuído;
- ***Map-Resolver:*** é um componente que recebe as mensagens de requisições de mapeamentos EID-RLOC (mensagens *Map-Request*) e as retorna caso não existam no banco de dados distribuído.

Um EID pode ser associado a mais de um RLOC através do **Mapeamento EID-RLOC**, com cada RLOC incluindo informações de **prioridade** e **peso**. O RLOC com o menor prioridade será associado aos pacotes destinados ao EID correspondente. Caso este RLOC se torne indisponível, o próximo RLOC com menor prioridade será associado aos pacotes (***failover***). Entretanto, quando dois ou mais RLOCs possuem o mesmo valor de prioridade, um **balanceamento de carga** será realizado entre estes RLOCs, distribuindo os pacotes de acordo com o valor do peso de cada endereço.

O protocolo LISP é utilizado para a troca de mensagens de controle entre roteadores LISP e o sistema de mapeamento LISP, como forma de prover uma comunicação entre *sites* LISP e *sites* não-LISP. Na Figura 3.6, pode-se visualizar o uso do protocolo LISP para prover comunicação entre dois *sites* LISP. Neste cenário, há dois *sites* LISP, A e B, com um cliente em cada *site* (1.0.0.1 e 2.0.0.2, respectivamente). Cada *site* LISP contém um xTR, que consiste em um roteador que funciona como ITR e como ETR, simultaneamente. Cada xTR contém um RLOC, sendo o do *site* A com RLOC 11.0.0.1 e do *site* B com RLOC 12.0.0.2. Os xTRs dos *sites* A e B devem, periodicamente, enviar mensagens *Map-Register* para o *Map-Server*, como forma de registrar os seus mapeamentos EID-RLOC. A cada mensagem *Map-Register* processada com sucesso, o *Map-Server* responde ao xTR com uma mensagem *Map-Notify*.

Caso o cliente do *site* A queira enviar um pacote para o cliente do *site* B, o primeiro envia o pacote para o xTR do *site* A (xTR-A) com os endereços de origem e destino sendo, respectivamente, 1.0.0.1 e 2.0.0.2, ou seja, os EIDs dos clientes. Ao chegar ao xTR-A, o mesmo funcionará como ITR, ficando responsável por mapear o EID de destino (2.0.0.2) para um RLOC de um do ETRs do *site* de destino. Para isso, o xTR-A verifica em seu cache LISP local se existe algum mapeamento EID-RLOC para o EID 2.0.0.2. Caso não exista, o xTR-A envia uma mensagem *Map-Request* para o *Map-Resolver* como forma de requisição para este mapeamento.

O *Map-Resolver*, caso possua esse mapeamento no banco de dados distribuído, responde à requisição com uma mensagem *Map-Reply*, contendo o mapeamento EID-RLOC para o endereço 2.0.0.2 (RLOC 12.0.0.2). De posse desse mapeamento, o xTR-A

armazena essa informação em seu cache LISP local e, em seguida, encapsula o pacote original com um cabeçalho LISP, incluindo o RLOC do *site* A como endereço de origem (11.0.0.1) e o RLOC do *site* B, obtido do mapeamento, como endereço de destino (12.0.0.2). Em seguida, o pacote é enviado em direção ao *site* B. O xTR do *site* B (xTR-B), ao receber o pacote, funcionará como ETR. Ele irá desencapsular o pacote, obtendo o cabeçalho IP mais interno com os endereços EID. Em seguida, irá encaminhar o pacote para o *host* que possui o EID igual ao EID de destino do pacote.

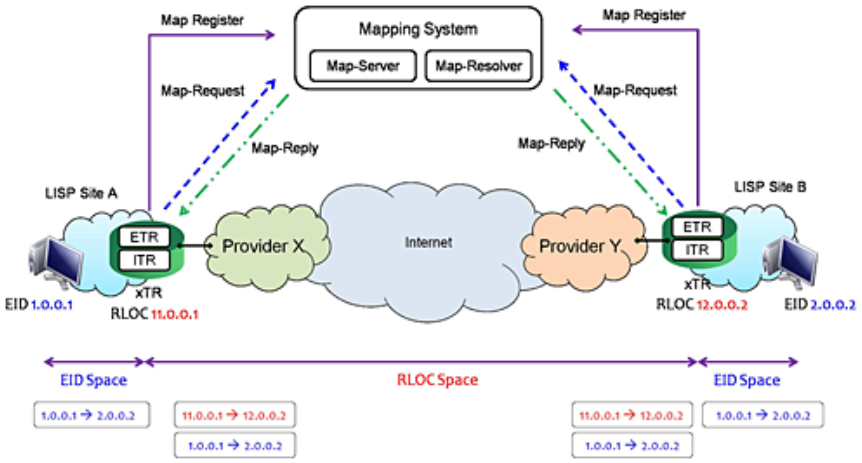


Figura 3.6. Arquitetura LISP. Fonte: [Jeong et al. 2015]

O LISP provê capacidade avançada de Engenharia de Tráfego por meio dos *Explicit Locator Paths* (ELPs). Um ELP serve como um mecanismo para forçar o tráfego por um determinado caminho. Ele consiste em uma lista de saltos pelos quais os pacotes devem ser roteados. Para a implementação dos ELPs, o LISP faz uso dos *Reencapsulating Tunnel Routers (RTRs)* [Kowal et al. 2015]. A Figura 3.7 ilustra o funcionamento de um RTR.

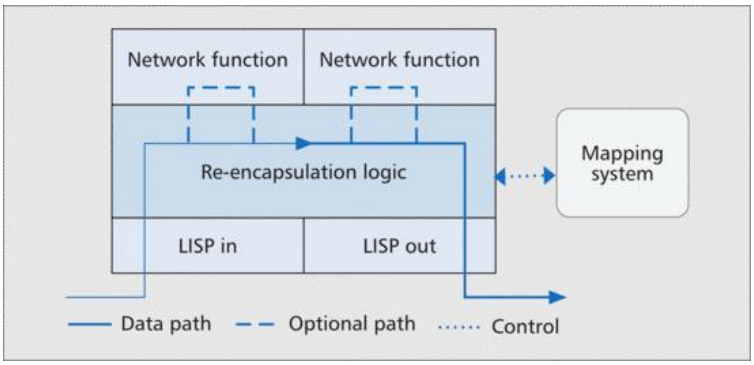


Figura 3.7. Arquitetura do RTR. Fonte: [Rodriguez-Natal et al. 2015]

O RTR é um roteador LISP localizado no núcleo da rede (espaço RLOC), ao contrário dos outros (ITR e ETR) que se encontram nas bordas da rede. No seu funcionamento, o RTR age como um ETR, desencapsulando pacotes cujo endereço de destino do

cabeçalho IP mais externo contém um dos seus RLOCS. Além disso, ele pode aplicar diferentes funções de rede no pacote desencapsulado, tais como: filtro de pacotes, sistemas de detecção/prevenção de intrusão, criptografia, análise de tráfego, qualidade de serviço, etc. Por fim, o RTR age como um ITR, tomando a decisão de como encapsular o pacote baseado no próximo salto do ELP existente no seu cache LISP local, ou obtido através do envio de uma mensagem *Map-Request* para o *Map-Resolver*, como forma de requisição para este mapeamento [Kowal et al. 2015]. Portanto, o ELP é uma lista de RLOCs para cada RTR que um pacote deve percorrer em direção ao seu destino final (ETR ou PETR). Os RLOCs devem ser percorridos na mesma ordem em que eles aparecem na lista do ELP.

Prioridades e pesos também são aplicados aos ELPs, o que significa que um EID pode ser mapeado para vários ELPs. Caso os ELPs possuam diferentes prioridades, o que tiver menor valor será utilizado por padrão para o fluxo de tráfego, enquanto os outros servirão de *backup*, caso o primeiro falhe (*failover*). Por outro lado, se múltiplos ELPs tiverem a mesma prioridade, um balanceamento de tráfego será realizado, com o fluxo de tráfego sendo dividido na proporção dos respectivos valores de peso.

Existe um interesse crescente na utilização do LISP como uma solução para SDN. Em [Kreutz et al. 2015], os autores incluem o LISP como uma das opções para a implantação de ***Overlay SDN***. Tal abordagem tem sido cada vez mais utilizada na virtualização de redes em *data centers* e comunicação *inter data centers*. Segundo os autores, *Overlay SDN* é uma solução baseada em *software* ou *hardware* em que a borda da rede é dinamicamente programada para gerenciar túneis entre hipervisores e/ou *switches*, criando uma rede sobreposta ou rede virtual, necessária para realizar a comunicação entre máquinas virtuais em ambientes *multi-tenant*. Tais túneis geralmente terminam em algum *gateway*, que podem estar representados por um *switch* virtual ou um dispositivo físico.

Em [Rodriguez-Natal et al. 2015], os autores avaliam se o LISP, na sua forma atual, pode ser usado como uma *API Southbound* para SDN. Para isso, foi feito um levantamento dos requisitos relevantes para SDN e como eles podem ser atendidos pela arquitetura LISP e seus componentes. Dentre os requisitos analisados estão a separação do plano de dados e de controle, a programabilidade da rede e o controle centralizado.

De acordo com os autores, o LISP realiza a separação dos planos de controle e de dados da seguinte forma:

- Plano de Controle: consiste no sistema de mapeamento LISP (*Map-Server* e *Map-Resolver*), sendo responsável por armazenar os mapeamentos EID-RLOC e as políticas de encaminhamento, bem como servir essas informações ao plano de dados;
- Plano de Dados: representado pelos roteadores LISP, sendo responsável pelo encapsulamento e desencapsulamento de pacotes trafegantes entre *sites* LISP ou entre um *site* LISP e não-LISP, de acordo com os mapeamentos EID-RLOC obtidos do plano de controle.

A programabilidade da rede é alcançada através da programação de políticas de encaminhamento, como mobilidade, cadeia de serviços e engenharia de tráfego, no sistema de mapeamento LISP. Por outro lado, o plano de dados poderá obter essa políticas por demanda. Além disso, desde que o sistema de mapeamento LISP armazena todos os

dados de estado da rede (mapeamento e políticas) que podem ser remotamente acessados e atualizados em tempo real, ele centraliza o controle da rede.

Finalmente, os autores também apresentaram os benefícios do LISP para a arquitetura SDN, como: a) escalabilidade, disponibilizada pelo sistema de mapeamento LISP que armazena informações do estado da rede, permitindo que os roteadores LISP recuperem e armazenem essas informações em cache local; b) interoperabilidade, pois permite trabalhar diferentes protocolos na camada de rede; e c) comunicação entre domínios, reforçada pelas capacidades avançadas de engenharia de tráfego do LISP.

Algumas ferramentas foram desenvolvidas para provimento de ambientes SDN através do uso do LISP, conforme descrito a seguir. O ***OpenDaylight*** (ODL) [ODL 2016] é uma plataforma SDN modular *open-source* mantida pela ***The Linux Foundation***[9]. Escrito em Java, o ODL tem como objetivo a aceleração no desenvolvimento de soluções para SDN e *Network Function Virtualization* (NFV) em ambientes de produção. Como forma de habilitar o desenvolvimento de ambientes SDN LISP, o ODL disponibiliza o serviço *LISP Flow Mapping*. Este serviço funciona como um plano de controle para os dispositivos LISP, provendo uma implementação do sistema de mapeamento LISP. Atualmente, o ODL encontra-se na versão *Beryllium*, lançada em Fevereiro de 2016.

Aplicações executadas sobre o ODL podem usar a *API Northbound* (REST) para definir mapeamentos EID-RLOC e políticas de encaminhamento (por exemplo, balanceamento de carga e engenharia de tráfego) no serviço de mapeamento LISP. Este serviço implementa o *Map-Server* e o *Map-Resolver*, que armazenam e disponibilizam dados de mapeamento para o plano de dados (dispositivos LISP), através do protocolo LISP disponibilizado por um *plugin* (API Southbound), ou para as aplicações ODL, através da *API Northbound*.

O ***Open Overlay Router*** (OOR) é uma ferramenta *open source* que permite criar redes *overlay*, utilizando o LISP. Escrito em C e distribuído sobre a licença Apache 2.0, o OOR está disponível para as plataformas Linux, Android e OpenWRT [OOR 2016]. Através do *LISPmob*, ele permite a criação de dispositivos LISP (ITRs, ETRs, RTRs, etc) e, combinado com o módulo *LISP Flow Mappings* do *OpenDayLight* [ODL 2016], redes *overlay* pode ser criadas, configuradas e disponibilizadas usando NETCONF/YANG [Enns et al. 2015] e gerenciadas pelo plano de controle LISP, em tempo de execução.

3.4. Engenharia de Tráfego em Redes Definidas por Software

Esta seção apresenta os principais protocolos e arcabouços de ET desenvolvidos com auxílio dos paradigmas de SDN. Novas técnicas, métodos e inovação são esperados no contexto de SDNs [Kreutz et al. 2015]. Conforme ilustrado na Figura 3.8, as abordagens que servem como base para ET são gerenciamento de fluxo, atualização de topologia, tolerância a falhas e caracterização/análise de tráfego, as quais serão detalhadas e devidamente referenciadas [Agarwal et al. 2013, Zhang et al. 2014]. A separação dos planos de controle e de dados leva a uma centralização do controle da rede. Esta abordagem pode levar a níveis elevados de congestionamento no controlador quando ele é submetido ao processamento de altas taxas de tráfego. Sendo assim, há a necessidade de desenvolvi-

[9]www.linuxfoundation.org/

mento de abordagens que otimizem a utilização da rede, reduzindo a perda de pacotes e atraso [Agarwal et al. 2013, Zhang et al. 2014], e que melhorem a forma de encaminhamento dos fluxos. As implementações em SDN estão definidas nos três planos da Figura 3.4.

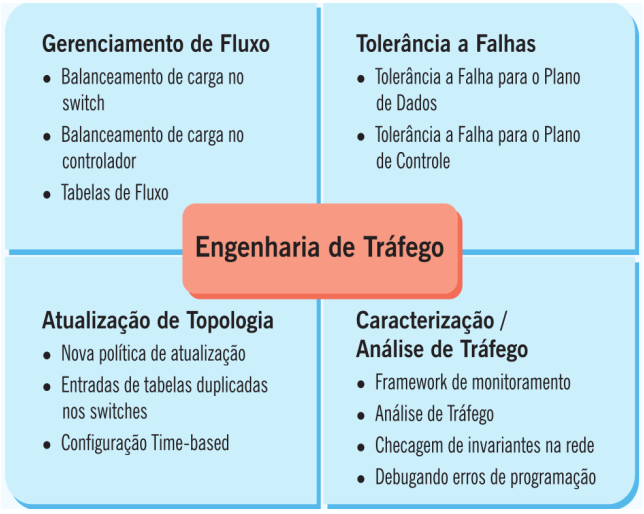


Figura 3.8. Escopo das abordagens de Engenharia de Tráfego em SDN. Fonte: [Akyildiz et al. 2014]

O trabalho [Akyildiz et al. 2014] descreve toda a evolução da ET, citando mecanismos relevantes e destacando as oportunidades de pesquisa de ET em SDN. Na literatura, há abordagens [Moshref et al. 2014, Zhang et al. 2014] que implementam o gerenciamento de fluxo no controlador, pois a *API Northbound*, ilustrada na Figura 3.4, não implementa tal funcionalidade. Novas abordagens de melhoria da ET que destacam as abordagens utilizadas em SDNs podem ser observadas em [Agarwal et al. 2013, Moshref et al. 2014, Yu et al. 2010].

A taxonomia, representada na Figura 3.9, indica trabalhos que foram selecionados através de critérios de relevância, impacto no meio acadêmico e na indústria de redes. As arquiteturas Hedera, MicroTE e B4 foram publicadas em conferências de prestígio e os artigos que as descrevem possuem muitas citações. As demais são mais recentes e também foram publicadas em conferências de prestígio. Ambos os cenários intra e inter *data center* têm impacto na geração de tráfego e apresentam problemas já descritos em seções anteriores. É importante destacar que os trabalhos citados não esgotam o estado da arte. Outros trabalhos de destaque na área podem ser encontrados em [Liu et al. 2016, Guo et al. 2014, Li et al. 2014] dentre outros.

3.4.1. Redes de *Data Center*

Segundo [Arregoces and Portolani 2003], as principais características de uma arquitetura de rede para *data center* são escalabilidade, flexibilidade e alta disponibilidade. Escalabilidade é a capacidade de manter um crescimento rápido no desempenho, número de dispositivos instalados no *data center* e a quantidade e a qualidade dos serviços ofertados. Quanto ao número de dispositivos, a escalabilidade refere-se à capacidade de adi-

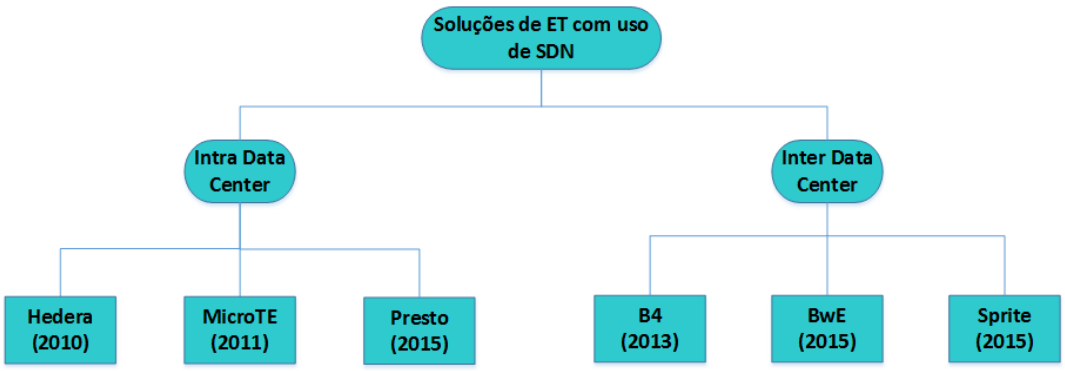


Figura 3.9. Taxonomia de Soluções de ET com SDN

cionar de forma simples servidores, roteadores, *switches* e *middleboxes* (exemplos de *middleboxes* são balanceador de carga, *firewall*, *Intrusion Detection System*, entre outros) [Arregoces and Portolani 2003].

No projeto de uma arquitetura de *data center*, flexibilidade é a capacidade de ofertar novos serviços sem exigir a reformulação completa da arquitetura. Uma proposta para oferecer flexibilidade é utilizar uma arquitetura modular, em que a adição de módulos com novos serviços seja uma tarefa simples. Alta disponibilidade se traduz em uma arquitetura redundante de forma a garantir que possíveis falhas sejam facilmente contornadas. Isso implica que o tempo de recuperação de qualquer falha deve ser menor que o tempo necessário para que o usuário perceba que a mesma ocorreu [Arregoces and Portolani 2003].

A topologia é um requisito de projeto fundamental para estruturar as redes de *data centers*. O tipo de topologia influencia diretamente no desempenho da rede no *data center* [Costa et al. 2012]. A topologia típica em *data centers* é formada por árvores hierárquicas compostas por *switches*. A seguir são brevemente descritas duas das principais topologias usadas em *data centers*:

- Fat-Tree: geralmente, essa topologia possui de dois a três níveis de elementos de encaminhamento. Em topologias com três níveis, os *switches* são classificados como de núcleo, de agregação e de acesso. Os servidores são conectados aos *switches* de acesso que, por sua vez, são conectados aos *switches* de agregação, os quais se conectam aos de núcleo. A topologia com dois níveis possui apenas os *switches* de núcleo e de acesso [Costa et al. 2012]. Essa topologia é apresentada na figura 3.10

- Clos: é uma topologia de rede que provê múltiplos caminhos entre a origem e o destino dos dados, possibilitando minimizar a probabilidade de o tráfego ser bloqueado. Os *switches* da camada intermediária possuem a mesma quantidade de enlaces ligando-os aos *switches* das camadas inferior e superior [Goransson and Black 2014]. A topologia *Fat-Tree* foi baseada na Clos, por isso essa última é similar à topologia apresentada na figura 3.10.

Essa abordagem de topologia em camadas é fundamental para que os *data centers* sejam capazes de prover escalabilidade, alto desempenho, flexibilidade, resiliência e faci-

lidade de manutenção [Verdi et al. 2010]. A topologia *Fat-Tree* utiliza *switches Ethernet* comoditizados com o objetivo de reduzir custos em *data centers*.

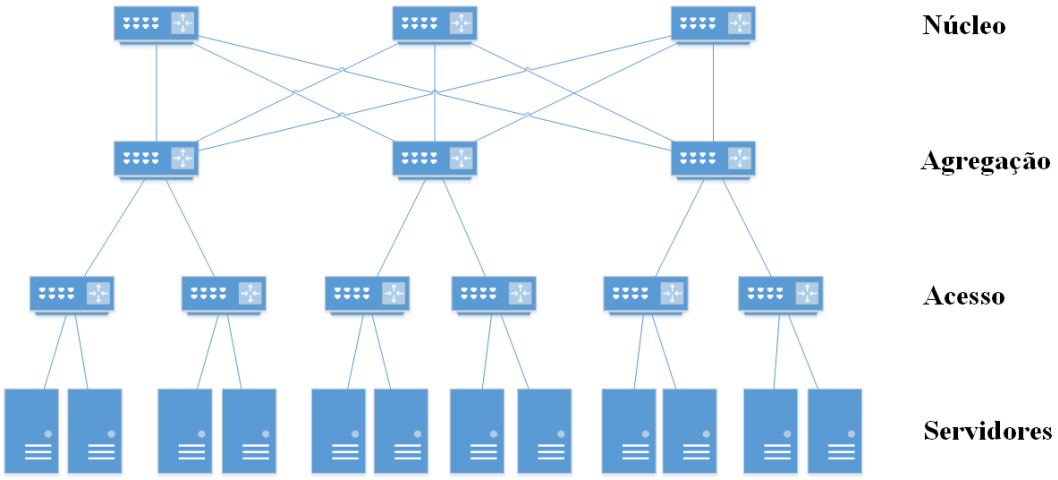


Figura 3.10. Topologia tipica de uma rede de *data center*. (Adaptado de: [Costa et al. 2012])

Geralmente, os servidores são montados em conjunto em um *rack* e interligados através de um *switch* de acesso. Esse *switch* é chamado de *Top-of-Rack* (ToR) e possui interfaces de 1 ou 10 Gbps para conectar-se com os *switches* de agregação. Os *switches* de agregação podem abranger mais de dez mil servidores individuais [Verdi et al. 2010]. Nos *data centers* em que a infraestrutura é virtualizada, cada servidor físico é capaz de hospedar múltiplas instâncias de servidores virtuais criados como máquinas virtuais através de *softwares hypervisores*. Nesse ambiente virtualizado, as máquinas virtuais em um servidor físico são interconectadas através de *switches* virtuais. Por exemplo, o *switch* virtual *Open vSwitch* foi desenvolvido com funcionalidades semelhantes às encontradas em dispositivos físicos, mas com a flexibilidade e velocidade de desenvolvimento ofertadas por implementações em *software*.

Um *data center* possui suporte a dois tipos de tráfego: aquele que entra e sai do *data center* e, o tráfego interno ao *data center* [Verdi et al. 2010]. Para receber requisições originadas na *Internet*, uma aplicação deve possuir um endereço IP válido. Geralmente, as requisições são atendidas por servidores chamados de *Front-Ends* que, comunicam-se com servidores *Back-Ends* para acessar serviços de armazenamento de dados distribuídos. Essa separação entre os servidores *Front-End* e *Back-End* aumenta a escalabilidade e simplifica o gerenciamento das aplicações ao permitir que partes dessas aplicações sejam implementas em servidores fisicamente distintos [Arregoces and Portolani 2003]. Em *data centers*, há dois tipos de tráfego. É importante identificar o tipo do tráfego para aplicar as técnicas de engenharia de tráfego apropriadas para cada tipo. A seguir são descritos esses tipos de tráfego:

- Tráfego elefante: é um fluxo de dados composto por uma quantidade de pacotes extremamente alta. Esse tipo de fluxo ocorre com uma frequência muito baixa.

Contudo, formam uma grande parcela do tráfego total, visto que são formados por grandes quantidades de dados em poucos fluxos [He et al. 2015].

- Tráfego ratos: fluxos formados por uma pequena quantidade de pacotes. Esse tipo de fluxo ocorre com maior frequência, mas representa uma pequena parcela do tráfego total devido à quantidade de pacotes nesses fluxos ser pequena [He et al. 2015].

As requisições que chegam da *Internet* são distribuídas entre os servidores através de um dispositivo chamado balanceador de carga. Esse dispositivo recebe as requisições e as distribui entre os servidores utilizando algoritmos como o *round-robin*[10]. O balanceador de carga possui uma lista com os endereços dos servidores para os quais as requisições recebidas deverão ser distribuídos. Esse dispositivo é transparente aos usuários que, creem estar acessando diretamente os servidores da aplicação requisitada.

3.4.2. Soluções para Ambientes Intra *Data Center*

Em topologias que possuem aumento no volume de tráfego Leste-Oeste (servidor para servidor ou servidor para *storage*) há um comprometimento da escalabilidade em Redes de *Data Center*. O ECMP (*Equal-cost Multi-path Routing*) é uma estratégia utilizada para balanceamento de carga em redes de *data center*. O funcionamento é baseado na atribuição aleatória dos fluxos entre caminhos existentes na rede. A principal desvantagem da estratégia é a possibilidade de colisão. O artigo de [Gharbaoui et al. 2014] realiza uma comparação da performance de diferentes algoritmos de ET para DC que podem ser adotados para serviços de nuvem.

3.4.2.1. Hedera

Um dos primeiros trabalhos a utilizar o *OpenFlow* para gerenciamento de fluxos em ambientes de *data center* foi [Al-Fares et al. 2010]. A solução propõe um sistema de escalonamento dinâmico de fluxos. O funcionamento do sistema ocorre através da coleta de informações de fluxos dos *switches*, calcula rotas não conflitantes para os fluxos e seta instruções para que os *switches* efetuem re-roteamento de acordo com as instruções setadas. O objetivo da proposta é maximizar a utilização da rede, através do conceito de largura de banda bisseção, que consiste na soma das capacidades de dois enlaces iguais, e fazer com que o escalonador tenha um menor atraso ou impacto nos fluxos ativos. Ao se adotar a possibilidade de ter uma visão global da rede, há a possibilidade da visualização do roteamento e das demandas de tráfego, tornando possível que o sistema de escalonamento visualize gargalos que o escalonador local dos *switches* não visualizam.

Representada em alto nível, a arquitetura realiza um laço de controle com três passos básicos:

1. Detectar fluxos grandes nos *switches* de borda;
2. Estimar a demanda por fluxos grandes e aplicar algoritmos de alocação para calcular boas rotas para os fluxos;

[10]Os servidores são dispostos em uma fila circular. O balanceador de carga redireciona a requisição para o servidor na primeira posição, e envia-o para o final da fila.

3. As rotas são setadas nos *switches*.

Arquitetura

O processo de inicialização do *switch* aproveita a utilização da topologia de árvore com várias raízes possibilitando o encaminhamento do tráfego entre quaisquer *hosts* o mais uniformemente possível entre todos os *switches* core. O caminho do pacote é não determinístico até o núcleo e determinístico quando retorna dos *switches* core ao seu *switch* de borda destino. Especificamente, para topologias de múltiplas raízes, há exatamente um caminho de custo mínimo ativo de qualquer *switch core* para qualquer *host* de destino [Al-Fares et al. 2010].

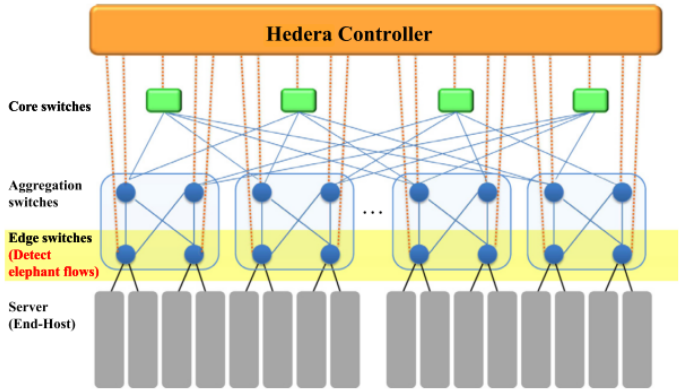


Figura 3.11. Arquitetura Hedera. Extraída de [Akyildiz et al. 2014]

A função do escalonador é alocar fluxos a caminhos não congestionados. Caso um fluxo persista e necessite de mais banda que o limite definido, será atribuído outro caminho de acordo com os seguintes algoritmos de escalonamento:

- *Global First Fit*: escalonador procura todos os caminhos possíveis nos quais seja possível acomodar o respectivo fluxo. Primeiro é realizada uma reserva de capacidade para o fluxo que será alocado no caminho correspondente. Segundo, o escalonador cria entradas de encaminhamento nos respectivos *switches* de agregação e borda;

- *Simulated Annealing*: executa uma busca probabilística para alocar de forma eficiente caminhos para os fluxos. Com a finalidade de reduzir o espaço de busca e consequentemente reduzir o esforço computacional, foi atribuído um *switch* de núcleo para cada *host* destino ao invés de *switch* de núcleo para cada fluxo. A escolha pelos dois algoritmos foi pela simplicidade. Algoritmos mais complexos podem ocasionar perda de eficiência do escalonador.

Os detalhes da implementação do Hedera podem ser resumidos em:

- **Topologia**: *fat-tree* conforme ilustrado na Figura 3.11 e características descritas na seção 3.4.1. O *testbed* consiste de 16 *hosts* interconectados por vinte *switches* de quatro portas. O escalonador de fluxos executa em uma máquina separada e é conectada a um *switch* de 48 portas;
- **Controlador *OpenFlow***: todos os *switches* na árvore executam *OpenFlow*. O funcionamento padrão do *switch* verifica se um pacote que chega não está associado a alguma entrada na tabela TCAM (*Ternary Content Addressable Memory*) ou SRAM (*Static Random Access Memory*), então o *switch* insere uma nova entrada de fluxo com a devida porta de saída (baseada no ECMP) o qual permite que qualquer pacote subsequente seja diretamente encaminhado na taxa permitida pelo hardware.

A Figura 3.12 indica a largura de banda bissessão alcançada para os padrões de comunicação utilizados. Pode-se concluir que o uso de um escalonador com conhecimento global da rede pode melhorar o balanceamento de carga baseado em *hash*. Vale destacar que o Hedera foi implementado sem a necessidade de nenhuma alteração na pilha de rede dos *hosts* finais ou sistemas operacionais.

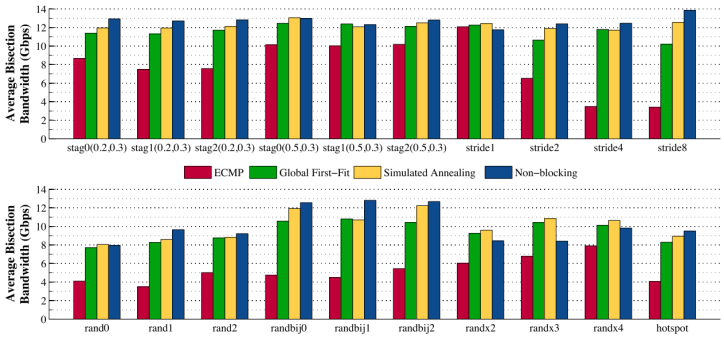


Figura 3.12. Resultados do *testbed* físico. Extraída de [Al-Fares et al. 2010]

3.4.2.2. MicroTE

A arquitetura MicroTE [Benson et al. 2011] foi implementada com objetivo de melhorar a sensibilidade (maior granularidade) na aplicação de ET. Esse *framework* em questão possui três componentes: monitoramento (monitora o tráfego e determina a previsibilidade entre os *racks*), roteamento (calcula as rotas baseado nas informações fornecidas pelo controlador) e o controlador (agrega a demanda de tráfego dos servidores e configura as rotas para os *switches*).

Arquitetura

Como observado na Figura 3.13, o componente de monitoramento no servidor evita que o controlador efetue *polling* nos *switches* e ocasione processamento no controlador. A abordagem com servidores permite as seguintes vantagens:

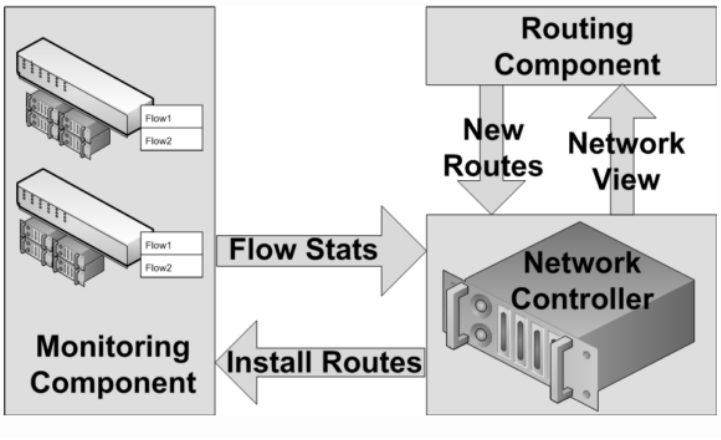


Figura 3.13. Arquitetura MicroTE. Extraída de [Benson et al. 2011]

1. O controlador receba atualizações da carga do tráfego, especialmente quando a carga muda significativamente. Na versão utilizada, o *OpenFlow* suporta somente *polling* pelo controlador. Essa solução é menos flexível;

2. Evitar que controlador gere quantidade significante de tráfego de controle na rede através de constantes *polling* em todos os *switches*;

3. Deslocar o gargalo da constante geração de estatísticas de fluxos dos *switches* para os *hosts* finais.

Além disso, cada servidor no componente de monitoramento captura o tráfego que está sendo enviado/recebido em suas interfaces. No entanto, somente um servidor no *rack* é responsável por agregar, processar e resumir as estatísticas da rede de todo o *rack*. O servidor, chamado de servidor designado, também envia a matriz de tráfego para o controlador. Para isso, o servidor designado deve executar as seguintes tarefas:

1. Coletar dados dos outros servidores no *rack*;

2. Agregar dados de servidor para servidor para dentro dos dados *Rack* para *Rack*;

3. Determinar pares ToR previsíveis;

4. Comunicar essa informação com o controlador da rede.

Na Figura 3.13, observa-se que o controlador recebe as informações descritas anteriormente, e as repassa para o componente de roteamento para que novas rotas sejam criadas. Além de receber informações dos monitores, o controlador também retransmite rotas recebidas do componente de roteamento para os *switches*. O controlador instala as entradas de fluxos apropriadas na tabela de encaminhamento de cada *switch* na rede.

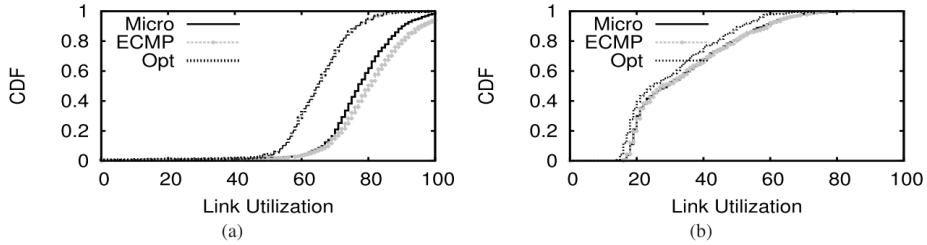


Figura 3.14. Extraída de [Benson et al. 2011]

3.4.2.3. Presto (*Proactive Load Balancing System*)

O sistema, baseado em *host*, proposto por [He et al. 2015] implementa balanceamento de carga quase ótimo de forma proativa em DCNs. A implementação tem o objetivo de reduzir o impacto negativo devido ao aumento do congestionamento ocasionado pelo desequilíbrio do tráfego na vazão, tempos de ida e volta e tempo de processamento dos fluxos. A proposta Hedera [Al-Fares et al. 2010], descrita na subseção 3.4.2.1, pelo fato de ser reativa ao congestionamento, ocasiona imprecisão devido ao alto custo computacional das estruturas de controle. O Presto executa em redes de 10+ Gbps, não necessita de hardware especial e modificação na camada de transporte nos *hosts* finais. Fluxos de tamanho arbitrário são transformados em pequenos sub-fluxos, formados por unidades de dados uniformes, que são distribuídos na rede de forma proativa e balanceada [He et al. 2015].

Projeto

A principal decisão do projeto foi a implementação do balanceamento de carga no *hypervisor* e no *vSwitch* (*switch* virtual), versão código aberto do Open vSwitch [OvS 2016], que estão na borda da rede. Vale destacar que, o *vSwitch* permite a modificação de pacotes sem a necessidade de alteração das VMs ou da camada de transporte. A bordagem também, modificou o GRO (*Generic Receive Offload*), implementada no *kernel* do Linux, para efetuar o reordenamento dos pacotes. Os algoritmos implementados no GRO devem ser escaláveis para execução em redes de alta velocidade. Fica evidente o paradigma de SDN aplicado na solução, pois foi migrada a implementação de balanceamento de carga em hardware, para software na borda da rede. Isso ocasionou desafios devido ao fato de que funcionalidades (balanceamento de carga) implementadas em hardware têm melhor desempenho que as implementadas em *software*. Além disso, a divisão em sub-fluxos podem ocasionar problemas, pois pacotes de um mesmo fluxo podem ser encaminhados por rotas distintas e ocasionar problema de entrega de *flowcells*[11] fora de ordem. O tamanho dos *flowcells* é baseado no TSO (*TCP Segment Offload*) e são de 64 KB. Detalhes do emissor e receptor abaixo.

O emissor é responsável por efetuar o balanceamento de carga de duas maneiras:

Global na borda da rede: utiliza um controlador centralizado para obter a to-

[11]Unidades discretas de pacotes geradas pelos *vSwitches* nas bordas da rede.

pologia da rede e transmitir a informação de balanceamento de carga para os *vSwitches* de borda. O principal objetivo dessa decisão é permitir que os *vSwitches* atuem no balanceamento de carga de forma conjunta para obter informações da topologia e matrizes de tráfego atualizadas [He et al. 2015]. O controlador divide a rede em conjuntos de múltiplas *spanning trees* e atribui para cada *vSwitch* um rótulo de encaminhamento em cada *spanning tree*.

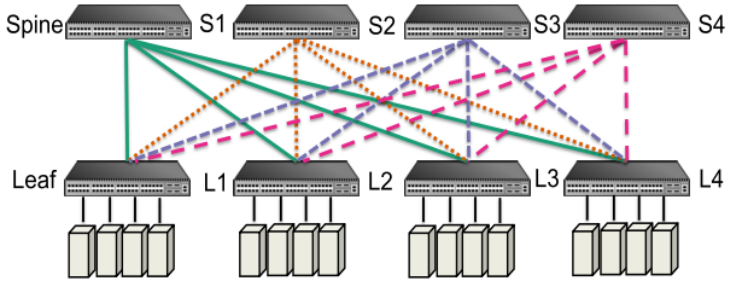


Figura 3.15. *Testbed* com uma rede Clos 2-*tier* . Extraída de [He et al. 2015]

A Figura 3.15 apresenta uma rede Clos 2-*tier* com 4 *switches spine* e 4 *spanning trees* distintas. Na existência de enlaces entre cada Spine e o *switch* folha (*leaf*), há possibilidade do controlador alocar *spanning trees* por *switch* Spine. Detalhes da topologia Clos 2-tier estão na subseção 3.4.1.

Balanceamento de Carga no Emissor: o controlador instala regras de encaminhamento baseadas na técnica de *Shadow* MACs (*Media Access Control*), proposto em [Agarwal et al. 2014], que é um tipo de comutação de rótulos para *Ethernet*. Essa técnica utiliza o endereço MAC de destino como um rótulo de encaminhamento opaco que pode ser instalado em tabelas de camada 2. Além disso, o controlador cria um mapeamento de cada endereço MAC físico de destino para uma lista correspondente de endereços *shadow* MAC. Isso possibilita um meio de enviar tráfego para um destino certo através de *spanning trees* distintas.

Receptor

A principal dificuldade do receptor é tratar o reordenamento, que eventualmente possa ocorrer, devido ao fato de que *flowcells* diferentes são enviados por rotas diferentes. O objetivo do receptor é reduzir os efeitos do problema de inundação de pequenos segmentos. A redução pode ser obtida por:

- não repassar de forma bruta para camada acima segmentos que não possam ser unidos com os pacotes que chegam
- assegurar que segmentos enviados são entregues em ordem.

3.4.2.4. Resumo Qualitativo Intra *Data Center*

A Tabela 3.1 apresenta uma breve descrição qualitativa das soluções de ET em SDN descritas nas subseções anteriores. Buscou-se uma variedade dos tipos de escopo e

abordagens como as indicadas na Figura 3.8 destacando-se as observadas na coluna 2.

Tabela 3.1. Resumo Qualitativo dos mecanismos de ET avaliados intra *data center*

Abordagem	Abordagem de ET	Distribuição dos controladores	Desvantagens
Hedera	Balanceamento de carga baseado em ECMP	Centralizado	Atraso de largura de banda nos *switches* é elevado
MicroTE	Monitoramento	Centralizado	Assumir a previsibilidade da Matriz de Tráfego
Presto	Balanceamento de Carga	Distribuído	Não considera congestionamento, ocasionando degradação do enlace em caso de falha

A arquitetura Hedera [Al-Fares et al. 2010] foi uma das pioneiras a utilizar SDN para balanceamento de carga em DCNs. Ambas as soluções (Hedera e MicroTE) têm problemas de desempenho por causa da alta volatilidade do tráfego em DCN. O mecanismo utilizado no Presto inovou e contribui em relação às duas propostas anteriores, pois utiliza uma abordagem proativa que lida com o reordenamento de pacotes sem gerar muito atraso no *host*. A desvantagem do Presto é não detectar congestionamentos.

3.4.3. Soluções para Ambientes Inter *Data center*

Os grandes *data centers* e provedores de serviços (ISP — *Internet Service Provider*) necessitam alcançar uma grande quantidade de usuários e garantir requisitos de QoS. Um dos principais objetivos dos DCs é otimizar o uso dos enlaces WAN. Em muitos casos esses enlaces são super provisionados para evitar congestionamentos. Isso ocasiona aumento no custo de manutenção dos DCs devido ao alto custo dos enlaces WAN. As técnicas de ET desenvolvidas no MPLS-TE não possibilitam a utilização de forma otimizada, pois não há uma coordenação central e cada provedor/*data center* estabelece a distribuição do tráfego de forma não determinística e sub-ótima [Jain et al. 2013]. A utilização de SDN possibilitou otimizar e flexibilizar técnicas de ET em WANs. Nessa seção serão detalhadas duas abordagens com o uso de SDN em WAN. As principais referências são: [Sun et al. 2015, Kumar et al. 2015]. Ao final da seção será inserido uma tabela com análise qualitativa das propostas revisadas. Serão analisados: abordagem proposta, breve descrição, tecnologia de ET e análise (resumo dos resultados).

3.4.3.1. B4

O artigo de [Jain et al. 2013] relata a experiência do Google no desenvolvimento e operação da WAN B4. O desenvolvimento foi fundamentado em princípios de SDN e *OpenFlow*. A rede foi planejada para operar com protocolos de roteamento e ET centralizada de forma simultânea. Os objetivos da ET na B4 são:

1. Permitir que a borda da rede efetue o controle e decida em caso de demanda por recursos em situação de escassez;
2. Utilizar encaminhamento/tunelamento multi-caminho para permitir disponibilidade de acordo com a prioridade da aplicação;
3. Dinamicamente realocar largura de banda em casos de falhas do *switch*/enlace.

Projeto

A arquitetura do B4 é disposta de forma lógica por três camadas, conforme ilustrado na Figura 3.16. Cada *site* WAN possui vários *clusters* situados na camada *switch hardware*. Essa camada não executa software de controle, tem a finalidade de encaminhamento do tráfego. A camada *site controller* é composta por NCS (Network Control Servers) que hospeda os controladores *OpenFlow* (OFC) e as Aplicações de Controle da Rede (NCAs). Os servidores possibilitam roteamento distribuído e ET centralizada como um *overlay* de roteamento. Os controladores (OFCs) mantêm o estado da rede baseados nas diretivas recebidas das NCAs e a configuração das entradas nas tabelas dos *switches* são baseadas na mudança do estado da rede.

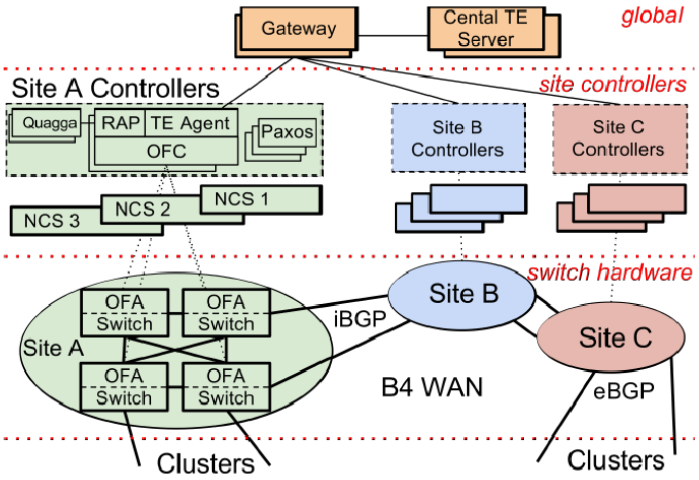


Figura 3.16. Visão geral da arquitetura do B4. Extraída de [Jain et al. 2013]

A camada Global é composta por aplicações logicamente centralizadas que habilitam o controle central de toda rede através dos NCAs. O *Gateway* SDN abstrai detalhes do OF e da camada *switch hardware* do servidor *central TE* conforme descrito em [Jain et al. 2013]. As aplicações da camada global são replicadas entre múltiplos *sites* WAN com eleição de líder separada para eleger o primário.

Os *switches* que operam em WANs geralmente possuem um *hardware* robusto (alta disponibilidade), boa quantidade de *buffers* e grandes tabelas de encaminhamento. Essas funcionalidades ocasionam um alto custo aos equipamentos. Os *switches* que executam nos *data centers* do Google não necessitam de grandes tabelas de encaminhamento, pois os *switches* funcionam em número reduzido de *data centers*. Por constatar que as falhas geralmente ocorrem no software, a possibilidade da customização do hardware reduzindo custos e o suporte a SDN foi então desenvolvido o *switch* ilustrado na Figura 3.17. A adoção desse hardware trouxe benefícios como ganho de eficiência e novos serviços como a ET centralizada.

O *switch* possui um sistema embarcado executando Linux. Um processo de nível de usuário foi implementado baseado em uma versão do OF com suporte a *pipeline* de

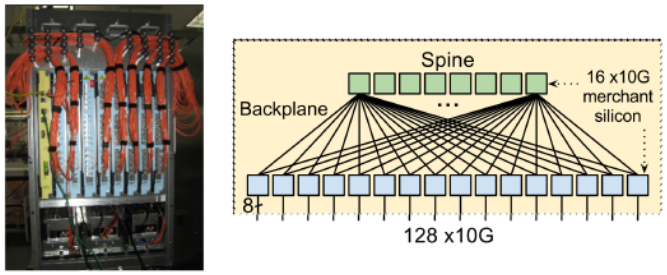


Figura 3.17. *Switch* customizado e sua topologia. Extraída de [Jain et al. 2013]

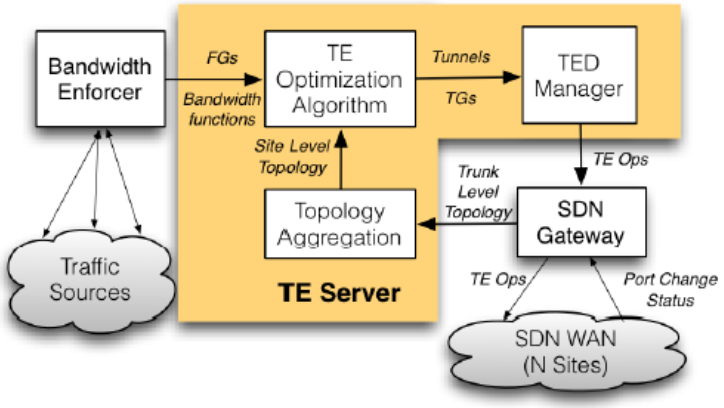


Figura 3.18. Visão Geral da Engenharia de Tráfego. Extraída de [Jain et al. 2013]

hardware e atuando como um agente OF (OFA - *OpenFlow Agent*). A função do OFA é traduzir mensagens OF em comandos para configurar entradas na tabela de encaminhamento.

O objetivo principal do B4 é efetuar alocação para aplicações através do algoritmo *max-min fairness*. Uma solução *max-min* maximiza a utilização, por exemplo de enlaces, enquanto minimiza a penalidade da parte "*fair*"(aplicações que não saturam o enlace). Um exemplo, supondo que numa rede com *switches* há enlaces entre origem e destino. Caso haja aumento no tráfego, de forma equânime, em todos os *switches* até que o primeiro enlace (ou enlaces) atinjam o saturamento. A partir daí, todos os *switches* que possuem enlaces para o enlace saturado não podem alocar qualquer outro fluxo (penalização), o enlace é desativado [Danna et al. 2012].

A arquitetura Centralizada de ET é visualizada na Figura 3.18. O Servidor de ET atua sobre os seguintes estados:

- O grafo **Network Topology** representa sites como vértices e as conectividades como arestas. O *Gateway* SDN consolida os eventos de *site*s e *switches* para o *Server*;

- **Flow Group (FG)**: para garantir escalabilidade o *Server* não pode operar a granularidade de aplicações de forma individual. Portanto, é realizada agregação para *Flow Group* definida pela tupla {*source site, dest site, QoS*};

- **Um Tunnel (T)** representa um caminho de *site*. B4 implementa túneis utilizando encapsulamento IP sobre IP;

- **Um Tunnel Group (TG)** mapeia FGs para um conjunto de túneis e pesos correspondentes.

3.4.3.2. BwE (*Bandwidth Enforcer*)

Um novo mecanismo para alocação de banda em WAN e com suporte a computação distribuída e transferência de dados foi apresentada em [Kumar et al. 2015]. Os roteadores atuam no processamento de pacotes e no mapeamento de políticas de alocação de banda. Essas funcionalidades nem sempre são atendidas, pois os roteadores não possuem capacidade para tratar, em situações de grande granularidade e grande volume de tráfego, pacotes individualmente. Seguindo o princípio fim-a-fim, a funcionalidade de mapeamento de políticas foi migrada para os *hosts* de origem [Kumar et al. 2015]. Os hosts possuem a capacidade de controlar o tráfego de saída e efetuar a marcação de pacotes através do campo DSCP (*Differentiated Services Code Point*) presente no cabeçalho do datagrama IP. Este campo é utilizado para encaminhamento e descarte de pacotes no caso de congestionamento. BwE cria uma hierarquia baseada nas informações de conhecimento global da topologia da rede e utilização do enlace. Essas informações são utilizadas pelos *global bandwitdth enforcers* e pelos *enforcers* que atuam em cada *host* para executar as políticas de alocação de banda.

O BwE é a principal solução de alocação de banda em WAN privada em produção. Dentre as contribuições destacam-se:

- Plano de controle hierárquico para gerenciamento de largura de banda extensível aos *hosts* finais, executando fora dos roteadores;

- Integração com outras soluções, inclusive a B4 descrita na subseção 3.4.3.1;

- O mecanismo de alocação de banda é *work-conserving* e flexível, possibilitando a implementação de uma série de políticas compartilhadas na rede.

Alguns conceitos e abstrações são importantes no funcionamento do BwE:

- *FlowGroups*: a distinção entre diferentes tarefas executando em um *host* é realizada através da modificação da pilha de rede do Linux, permitindo a marcação

Conforme ilustrado na Figura 3.19, a alocação de banda e a política de marcação dos pacotes, atuam de cima para baixo e a demanda dos fluxos dos *hosts* de baixo para cima. Os principais componentes são [Kumar et al. 2015]:

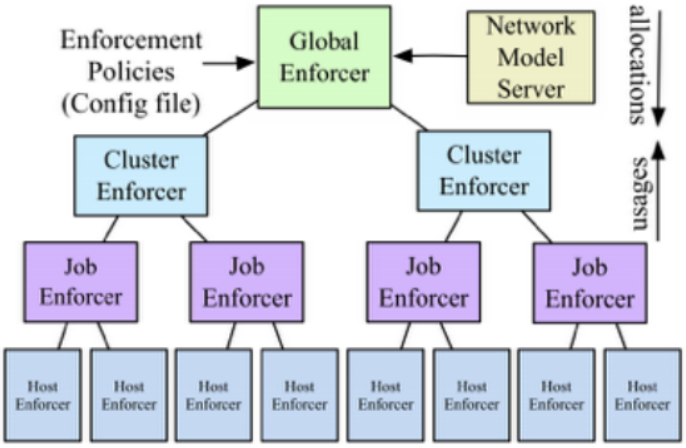


Figura 3.19. Arquitetura BwE. Extraído de [Kumar et al. 2015]

- ***Host Enforcer***: executa na forma de *daemon* no espaço de usuário nos *hosts* finais. Está no nível mais baixo da arquitetura. Aloca largura de banda baseada na disciplina de serviço HTB (*Hierarchical Token Bucket*) implementada no *kernel* do Linux.

- ***Job Enforcer***: agrega usos de *task-fgs* (unidade de alocação de largura de banda) para *job-fgs* (uso de largura de banda utilizada por todos *task-fgs* pertencentes ao mesmo *job*) e relatam usos *job-fgs* a cada 10 segundos para o *Enforcer Cluster*.

- ***Cluster Enforcer***: gerencia dois níveis de agregação de *user-fgs* para *jobs-fg* e *cluster-fgs* (largura de banda utilizada por todos *users-fgs* pertencentes ao mesmo grupo de usuários arbitrário) para *users-fgs*.

- ***Cluster Enforcer***: gerencia dois níveis de agregação de *user-fgs* para *jobs-fg* e *cluster-fgs* para *users-fgs*. Agrega as funções de alocação de banda *user-fgs* (cobrindo sua demanda estimada) para funções *cluster-fgs*, reportando-as para o *Global Enforcer* a cada 15 segundos.

- ***Network Model Server***: constrói o modelo de rede abstrato para o BwE. As informações da rede são coletadas via mecanismos de monitoramento, e.g. SNMP (*Simple Network Management Protocol*).

- ***Global Enforcer***: divide a capacidade de largura de banda disponível na rede entre *clusters* diferentes.

A arquitetura do BwE possibilita que a WAN alcance bons níveis de utilização. Muitos dos enlaces atingem 90% de utilização [Kumar et al. 2015].

3.4.3.3. Sprite

A Solução de [Sun et al. 2015], denominada de Sprite *Scalable Programmable Inbound Traffic Engineering*, controla o tráfego de entrada dividindo os endereços IP públicos da borda da rede entre ISPs. A técnica de SNAT (*Source Network Address Translation*) é utilizada para mapear cada conexão de saída para uma entrada específica para tráfego de retorno no ISP. A principal contribuição do Sprite é a implementação de ET, de forma escalável e alto nível (próxima do usuário), através de SDN. A escalabilidade no plano de dados é alcançada através da distribuição da funcionalidade do SNAT nos *switches* de borda mais próximos aos usuários finais. No plano de controle, agentes locais instalam regras de SNAT e coletam estatísticas de tráfego que são enviadas para o controlador central. O administrador da rede especifica uma meta de ET de alto nível baseada em nomes e métricas de performance. O controlador traduz a meta de alto nível em uma política de rede, e dinamicamente ajusta a política de rede baseada no tráfego e medidas de performance.

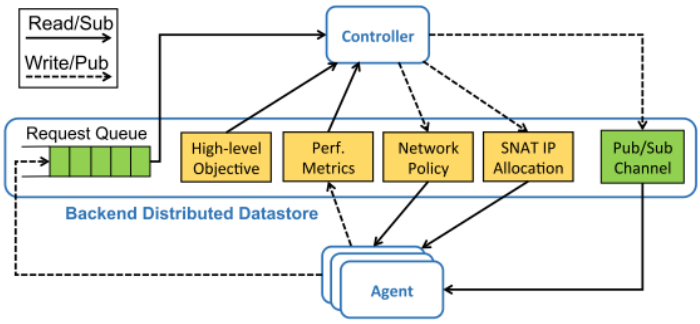


Figura 3.20. Arquitetura Sprite. Extraída de [Sun et al. 2015]

A Figura 3.20 ilustra a arquitetura do Sprite. Todos os dados são armazenados de forma distribuída e todas as informações de estado relacionadas ao objetivo de alto nível, política de rede, métricas de performance das conexões SNATed e o status de alocação do IP SNAT são mantidas. O controlador e todos os agentes executam de forma independentemente sem manter estado. Não há comunicação direta entre eles, somente leitura e escrita de dados através de armazenamento distribuído.

O controlador utilizado na solução é o *Floodlight* que interage com os agentes através do armazenamento de dados de forma *pull-based*. O método de comunicação *push-based* é utilizado para para balancear a performance do sistema. Através de dois pontos de comunicação no *Datastore* para sinalização *push-based* entre o controlador e os agentes: um canal *publish/subscribe* e uma fila de mensagem, como ilustrado na Figura 3.20 [Sun et al. 2015]. Em redes *OpenFlow*, o Sprite instala regras de roteamento para direcionar os pacotes que retornam aos agentes, ou seja, uma vez que a faixa de endereço IP/porta de origem é atribuída a um agente, o controlador instala regras *OpenFlow* para associar (efetuar um "*match*") na faixa IP/porta dos *switches* do roteador de borda para o agente. Ao invés de instalar uma regra por endereço IP de origem alocado nos *switches*, buscou-se consolidar as regras de roteamento através de um *"matching"* em um bloco de

prefixo maior. O algoritmo funciona da seguinte forma: uma árvore de menor caminho é criada com o roteador de borda na raiz com todos os agentes nas folhas. Quando um endereço IP de origem é alocado para um agente é escolhido o que é mais próximo dos IPs alocados para os agentes que tem o maior caminho compartilhado.

3.4.3.4. Resumo Qualitativo Inter *Data Center*

As soluções, apresentadas na subseção anterior, têm o foco em ET para WAN. Todas foram implementadas no Plano de Controle como ilustrado na Figura 3.4.

Tabela 3.2. Resumo Qualitativo dos mecanismos de ET avaliados inter *data center*

Abordagem	Abordagem de ET	Distribuição dos controladores	Análise
B4	Realiza coleta de estatísticas dos switches no controlador e efetua alocação de recursos e encaminhamento multi-caminho baseado nessas estatísticas. Utiliza uma versão customizada do ECMP para balanceamento de carga	Centralizado	Próximo de 100% de utilização da maioria dos enlaces. Média de 70% de utilização em longos períodos.
BwE	Mecanismo de alocação de banda utilizado em grande escala	Centralizado	Alcança 90% de utilização dos enlaces.
Sprite	Mecanismo escalável que realiza ET no tráfego de entrada em ISPs	Centralizado	A solução apresentou bom desempenho com testes realizados com vídeos

O trabalho de [Alizadeh et al. 2014] implementou o balanceamento de carga no Plano de Dados. Isso possibilitou um melhor tempo de resposta na tomada de decisão, na ordem de poucos microsegundos. Para isso, foi necessária a implementação de um *hardware* especial e proprietário, o qual inviabiliza modificações no algoritmo de balanceamento de carga. Essa abordagem, apesar de melhorar o balanceamento de carga, é contrária ao paradigma de SDN e apresenta as desvantagens citadas anteriormente. Como descrito na subseção 3.4.3.1 o Google desenvolveu um *switch* customizado, que apesar de não estar disponível para o mercado, há detalhes de sua implementação no trabalho [Jain et al. 2013].

3.5. Estudos de Caso

Nesta seção, são abordados os experimentos realizados através da configuração e a execução de dois estudos de caso. Estas práticas utilizam diferentes tecnologias SDN (*OpenFlow* na subseção 3.5.1 e LISP na subseção 3.5.2) para implementar soluções de engenharia de tráfego, em ambientes específicos. Todos os experimentos são detalhados e disponibilizados para que o leitor possa reproduzi-los.

3.5.1. Emulando uma aplicação de Engenharia de Tráfego em SDN com o Mininet

Esta subseção apresenta um estudo de caso envolvendo o emulador de redes SDN *Mininet* [Lantz et al. 2010] e os passos necessários para a realização do experimento de ET descrito em [Nguyen and Kim 2015] nesse *software*. São brevemente apresentadas a CLI (*Command-Line Interface*) e API *python* empregadas pelo *Mininet* na construção de uma rede SDN com todos os seus componentes apresentados na Figura 3.4.

Além disso, será descrito o algoritmo apresentado em [Nguyen and Kim 2015], em que foram empregadas técnicas de ET como alocação de fluxos e balanceamento de carga em redes SDN para calcular o caminho mais curto entre a origem e o destino do tráfego.

3.5.1.1. Construção de uma rede SDN no *Mininet*

O *Mininet* é um *software* de emulação para prototipação e experimentação de redes SDN. O *Mininet* é empregado na criação de redes compostas por *hosts* virtuais, *switches*, roteadores, controladores e enlaces [Keti and Askar 2015]. O *Mininet* possui uma CLI e uma API *Python* usadas na construção da rede para experimentação.

O comando *mn* é utilizado para iniciar a CLI do *Mininet*. Com o comando *mn* é possível criar redes com topologias pré-definidas ou customizadas; definir características dos enlaces como: largura de banda e atraso; especificar o tipo de controlador e *switch* (além do *Open vSwitch*, o *Mininet* possui suporte a um *switch* virtual que atua como um *layer* 2 padrão); executar testes de desempenho com o *iperf*; dentre outros. Encontram-se disponíveis na página *web*[12] do *Mininet* exemplos dos comandos de sua CLI.

O *Mininet* possui uma API *Python* para construção de topologias customizadas de acordo com as necessidades do pesquisador. A descrição completa da API do *Mininet* pode ser encontrada em sua página *web*[13]. Além disso, vários exemplos de redes construídas com essa API estão disponíveis na página do *Mininet* no *github.com*[14].

O *Mininet* permite especificar o desempenho dos enlaces e *hosts* da rede, como a largura de banda e o atraso dos enlaces, e o limite (em porcentagem) que um *host* pode utilizar da CPU. Há exemplos da especificação do desempenho no *Mininet* disponíveis em sua página no *github.com*.

3.5.1.2. Execução de um experimento de engenharia de tráfego

Algoritmos de SPF (*Shortest Path First*) [Caesar et al. 2005] são amplamente utilizados nas redes atuais [Agarwal et al. 2013]. Contudo, esses algoritmos transmitem todo ou a maior parte do tráfego pelo mesmo caminho, o que leva ao congestionamento desse caminho e a má utilização dos recursos da rede. Esse problema ocorre porque algoritmos como o OSPF utilizam o caminho com menor custo, que é sempre o mesmo caminho [Nguyen and Kim 2015]. Algoritmos de SPF não são flexíveis o suficiente para ter suporte a ET em uma rede com diferentes aplicações [Akyildiz et al. 2014]. Teoricamente esse problema poderia ser resolvido por algoritmos de múltiplos caminhos como o ECMP. Contudo, esses algoritmos podem utilizar caminhos com condições de tráfego muito diversas de forma a inviabilizar o funcionamento de aplicações de tempo real sensíveis ao atraso e ao *jitter*, como telefonia VoIP (*Voice over IP*). Mais detalhes sobre o OSPF e o ECMP podem ser encontrados na Seção 3.2.

Em [Nguyen and Kim 2015], os autores utilizam a visão global do controlador SDN sobre a rede que gerencia para calcular o SPF e alocar os fluxos para diferentes caminhos entre a origem o destino dos dados. Considerando os desafios em calcular o SPF, os autores desse artigo avaliaram como aplicar ET na implementação de um algoritmo para calcular o SPF em SDN. No desenvolvimento desse algoritmo, foram empregadas

[12] http://mininet.org/walkthrough/

[13] http://mininet.org/api/hierarchy.html

[14] https://github.com/mininet/mininet/tree/master/examples

técnicas de ET em redes SDN como alocação de fluxos e balanceamento de carga. O algoritmo proposto considera a carga total dos enlaces para calcular o caminho. Dessa forma, os fluxos são sempre alocados para enlaces com a menor carga de tráfego. Assim, a probabilidade de ocorrer congestionamento é menor e o atraso fim a fim é reduzido.

Esse algoritmo garante que o tráfego seja balanceado entre diferentes caminhos, garantindo uma melhor utilização dos recursos da rede. Além disso, é possível prover QoS, visto que ao priorizar o tráfego de determinadas aplicações esse algoritmo poderá selecionar o caminho que melhor atenda aos requisitos da aplicação priorizada.

Esse algoritmo é chamado de *Accumulative-Load Aware Routing* (ALAR). O ALAR foi desenvolvido no controlador SDN *OpenIRIS*[15]. Ao calcular o SPF entre a origem e o destino dos dados em uma rede SDN, o ALAR considera a carga total de cada enlace que compõe o caminho. O algoritmo ALAR utiliza o módulo *Link Discovery* do *OpenIRIS* para obter as informações sobre o tráfego nos enlaces. O protocolo *OpenFlow* foi utilizado no controle e gerenciamento do tráfego como apresentado na Figura 3.1. Assim, o controlador gerencia os *switches* para que encaminhem o tráfego de sua origem até o seu destino.

O algoritmo ALAR atribui um custo para cada enlace da rede com base na quantidade de tráfego presente nesse enlace. Aos enlaces com muito tráfego é atribuído um alto custo, com o objetivo de desestimular a instalação de novos fluxos. Por outro lado, enlaces transferindo pequenas quantidades de dados possuem um custo menor. Portanto, esses enlaces têm mais chances de transferir os dados. O objetivo dessa proposta é melhorar a utilização dos recursos da rede, minimizar o congestionamento e reduzir o atraso fim a fim. A função que calcula o custo de um enlace é formulada da seguinte forma:

$$C = \frac{B_{ref}}{B_L}(C_{init} + \sum_i c_i.n_i),$$

em que a carga total do enlace é o somatório dos diferentes tipos de fluxos. c_i é o custo do fluxo i e n_i é a quantidade de fluxos i no enlace. Por exemplo, assumindo que os custos de fluxos HTTP (*Hypertext Transfer Protocol*) e FTP (*File Transfer Protocol*) são $c_{http} = 2$ e $c_{ftp} = 4$, respectivamente. E, há oito (n_i) fluxos HTTP e cinco (n_i) FTP instalados no enlace, a carga total no enlace será $8 \times 2 + 5 \times 4 = 36$. A variável C_{init} é usada para tornar as decisões de roteamento mais estáveis no decorrer do tempo e seu valor é configurado pelo administrador da rede. A fração da largura de banda (B_{ref}/B_L) faz com que enlaces com maior largura de banda tenham um custo menor para instalar novos fluxos.

O experimento descrito em [Nguyen and Kim 2015], em que o *Mininet* é utilizado para construção de uma rede experimental para avaliação de um mecanismo de ET para redes SDN, foi executado para avaliar o ALAR. Nessa avaliação, foi utilizada uma topologia *grid* apresentada na Figura 3.21. Segundo os autores do ALAR, a topologia *grid* foi escolhida por oferecer inúmeros caminhos entre a origem e o destino dos dados e que dessa forma podem facilmente examinar o desempenho de seu algoritmo de roteamento quando a carga na rede muda continuamente. Foram medidos o atraso e a largura de banda disponível entre os dois *hosts* da rede apresentada na Figura 3.21. O programa

[15]http://openiris.etri.re.kr/

ping é utilizado para medir o atraso, enquanto o *iperf* é usado para avaliar a largura de banda disponível.

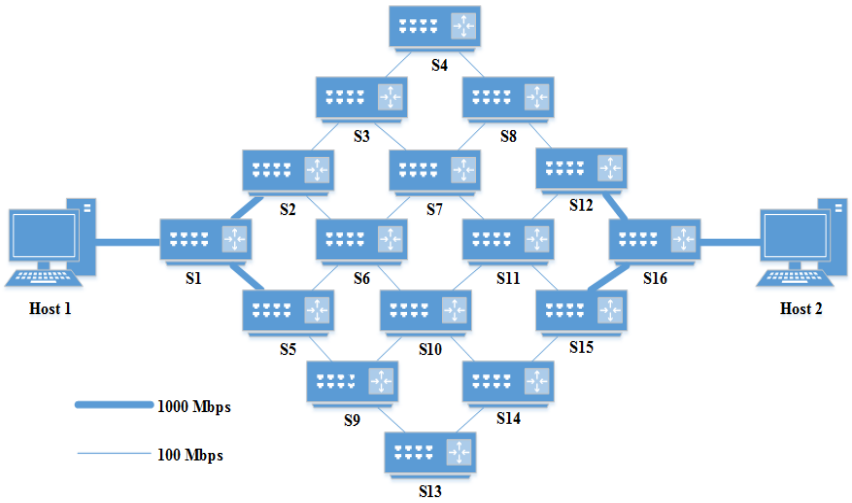


Figura 3.21. Topologia utilizada na avaliação do algoritmo ALAR. Adaptada de: [Nguyen and Kim 2015]

O ALAR[16] e o cenário[17] construído no *Mininet* para avaliação desse algoritmo estão disponíveis na *Internet* para *download*. Na reprodução desse experimento foi utilizada uma máquina com 4GB de memória RAM, processador *Intel core* i3-3217U de 1,8GHz com dois núcleos e sistema operacional *Ubuntu Linux* 14.04.4 LTS (*Long Term Support*). A seguir são apresentados os passos necessários à reprodução desse experimento:

O primeiro passo é instalar os *softwares* necessários à execução desse experimento:

```
$ apt-get install unrar openjdk-7-jre
```

em seguida é necessário baixar os arquivos do ALAR e o cenário do *Mininet*. Depois deve-se iniciar um terminal e acessar a pasta onde esses arquivos foram salvos. Os arquivos do ALAR vêm compactados no formato *.rar* que, pode ser descompactado utilizando o *software* anteriormente instalado *unrar*. Descompactar os arquivos do ALAR:

```
$ unrar x ALAR_Controller.rar
```

serão criadas duas pastas, o ALAR está na pasta *Torpedo*. Acesse essa pasta. O *OpenIRIS* é implementado em *JAVA*, por isso foi necessário instalar o *Java Runtime Environment* (JRE) para possibilitar sua execução. Executar o algoritmo ALAR:

```
$ sudo java -jar Torpedo.jar
```

[16]Disponível em: http://bit.ly/1LZaieM
[17]Disponível em: http://bit.ly/1Re1ynl

O cenário do *Mininet* foi disponibilizado como um *script python* chamado *scenario.py* que, ao ser executado realiza o experimento e salva os resultados em arquivos do tipo texto em uma pasta chamada *output*. Crie a pasta *output* na mesma pasta onde está o arquivo *scenario.py*. Execute o *script scenario.py*. Ao final da execução do *scenario.py*, os resultados sobre o tempo de resposta e a largura de banda estarão disponíveis na pasta *output*.

Com os resultados do ALAR obtidos, o próximo passo é executar o algoritmo SPF[18] padrão do controlador *OpenIRIS* e comparar seu desempenho com o do ALAR, como foi realizado em [Nguyen and Kim 2015]. Os passos para executar esse algoritmo são os mesmos seguidos na execução do ALAR. Por exemplo: baixar e descompactar o algoritmo, acessar a pasta *Torpedo* e executar o *Torpedo.jar*, criar a pasta *output* e executar o *scenario.py*.

As Figuras 3.22 e 3.23 apresentam um comparativo entre o algoritmo SPF padrão do *OpenIRIS* e o *ALAR* tendo como métricas a largura de banda e o tempo de resposta. Esses resultados demonstram que o ALAR apresenta um desempenho superior ao SPF padrão do *OpenIRIS*. A largura de banda disponível com o algoritmo padrão cai mais rapidamente devido à utilização do mesmo caminho para transferir todo o tráfego. Por outro lado, o ALAR distribui o tráfego entre diferentes caminhos, o que contribui para uma redução menor na largura de banda. Quanto ao tempo de resposta, como o ALAR aloca os fluxos para diferentes caminhos isso reduz o congestionamento dos enlaces e contribui para a redução do tempo de resposta.

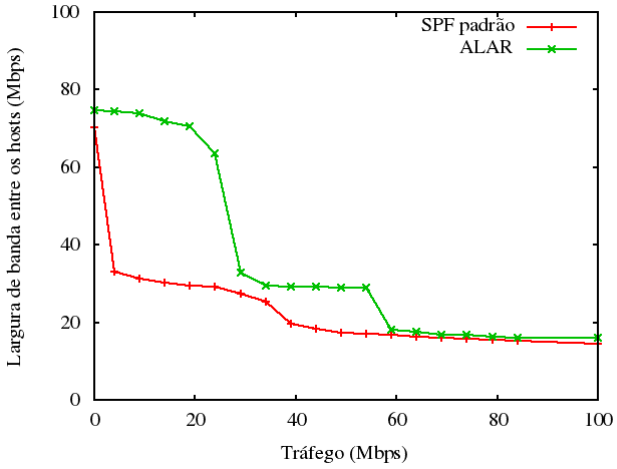


Figura 3.22. Largura de banda entre os *hosts* h1 e h2.

Com o *Mininet* é possível desenvolver o cenário utilizado na avaliação de uma solução como um *script python*, automatizando completamente o processo de execução de um experimento. E, com a conclusão do experimento, obtêm-se os resultados a serem analisados e divulgados. Assim, além de tornar a avaliação mais barata em relação ao uso de uma rede física, o *Mininet* simplifica muito a execução de experimentos que requerem redes SDN, visto que o experimento pode ser completamente automatizado.

[18]Disponível em: https://github.com/openiris/IRIS/archive/v2.2.1.zip

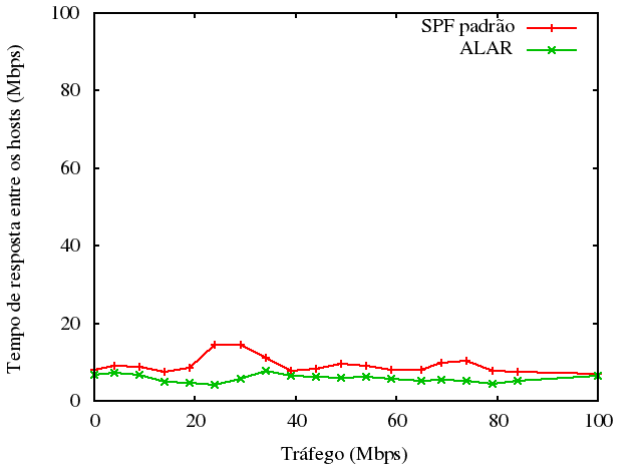


Figura 3.23. Tempo de resposta entre os *hosts* h1 e h2.

Nesta seção, foi apresentado o emulador de redes SDN *Mininet*, realizada uma breve descrição de sua CLI e API para construção de uma rede experimental e a execução de um experimento de ET em redes SDN. O *Mininet* possibilita a emulação de redes em que podem ser executados experimentos para avaliar novas soluções de ET para SDN.

3.5.2. Engenharia de Tráfego com o Protocolo LISP - Experimentação na plataforma FIBRE

Neste estudo de caso, utilizaremos o módulo LISP do *OpenDayLight* para implementar e explorar técnicas de engenharia de tráfego em redes *overlay* SDN com LISP. Na figura 3.24 é possível visualizar o cenário utilizado, sendo a topologia baseada no trabalho de [Rodriguez-Natal et al. 2015].

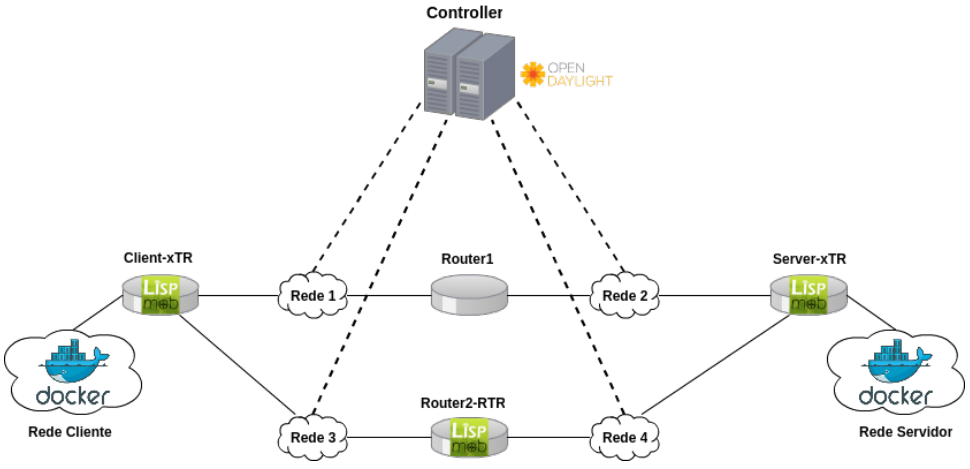


Figura 3.24. Topologia do cenário de testes de engenharia de tráfego com LISP.

Os dispositivos **Client-xTR** e **Server-xTR** funcionam como roteadores de borda realizando tanto as funções do ITR como do ETR (definidos na seção 3.3.3), mantendo

assim dois *sites* LISP. Ambos estão conectados a duas redes externas: redes 1 e 3 no caso do primeiro e 2 e 4 no segundo. O **Router2-RTR** realizará o papel de RTR na rede de trânsito, estando conectado às redes 3 e 4. Para viabilizar o funcionamento do plano de dados LISP, será utilizada a ferramenta LISPmob [OOR 2016] nos três dispositivos acima.

Por outro lado, o dispositivo **Controller** representa o plano de controle para os dispositivos LISP provendo uma implementação do Sistema de Mapeamento LISP, através do serviço *LISP Flow Mapping* do *OpenDayLight* (ODL). Para tanto, ele deve se conectar às quatro redes existentes, para que possa alcançar e ser alcançado por todos os outros dispositivos (xTRs e RTR). O **Router1** funciona apenas como um roteador IP simples, realizando apenas o trabalho de encaminhamento de pacotes entre as redes 1 e 2.

Além disso, os dispositivos que funcionarão como cliente e servidor nos experimentos são criados através de *containers Docker*[19], instalados respectivamente nas máquinas **Client-xTR** e **Server-xTR**.

Podemos visualizar que existem dois caminhos (*paths*) possíveis para o tráfego gerado entre os xTRs, sendo que em um deles temos um RTR intermediando. Este cenário permite a realização de experimentos com os mecanismos de controle de tráfego do LISP, incluindo balanceamento de carga e *failover*, bem como engenharia de tráfego através da definição de ELPs (*Explicit Locator Paths*), todos definidos na seção 3.3.3.

A seguir, descrevemos todo o processo de instalação e configuração do nosso cenário (subseção 3.5.2.1) e, em seguida, um relato das ferramentas necessárias e de como proceder com os experimentos de engenharia de tráfego com LISP (subseção 3.5.2.2).

3.5.2.1. Instalação e Configuração do Cenário de Experimentação

O cenário deste experimento foi criado e configurado sobre o ambiente para experimentação FIBRE [FIBRE 2016]. O *testbed* FIBRE consiste em um laboratório virtual para estudantes pesquisadores que desejam testar novas aplicações e modelos. Para tanto, ele disponibiliza recursos para a virtualização de dispositivos computacionais (máquinas virtuais), experimentação com redes *OpenFlow* e com redes WiFi. As duas primeiras funcionalidades são providas através do *OFELIA Control Framework* (OCF).

Atualmente, a infraestrutura do ambiente FIBRE é composta por 11 *clusters* locais (e 4 planejados), também conhecidos como "ilhas". Cada ilha possui sua própria infraestrutura física formada por dispositivos de rede e servidores, que suportam experimentos com redes *OpenFlow* (todas as 11 ilhas) bem como redes Wi-Fi (apenas 5 ilhas). Além disso, cada ilha possui a sua própria ferramenta de gerência OCF.

O OCF trabalha com o conceito de Projeto. Cada Projeto tem a seu dispor um determinado conjunto de recursos. Dentro de cada projeto, o usuário pode reservar recursos em geral virtualizados para uso no seu experimento, que incluem funções como: virtualização de dispositivos (máquinas virtuais) e redes (usando *OpenFlow*). Para cada experimento criado, o usuário poderá criar máquinas virtuais e conectá-las através de um

[19] docker.com

FlowSpace, uma rede virtual criada com *OpenFlow* e que pode ser segmentada com o uso de VLANs[FIBRE 2016].

Todas as ilhas são conectadas através de uma rede *overlay* criada sobre o *Backbone* da RNP, chamada de FIBREnet[FIBRE 2016]. Cada instituição participante tem autonomia sobre seus recursos (ilha) locais. Entretanto, eles também podem usar recursos de múltiplas ilhas para configurar os seus experimentos, através de uma instância central do OCF, chamada NOC.

A seguir, descreveremos os passos necessários para a criação do nosso cenário de experimentação, são 5 ao todos:

Passo A Criação do Cenário de Testes no Ambiente FIBRE;

Passo B Configuração das Interfaces de Rede;

Passo C Instalação e Configuração do *Ryu Framework*;

Passo D Instalação e Configuração do *OpenDayLight*;

Passo E Instalação e Configuração do LISPmob;

Passo F Instalação e Configuração do *Docker*.

Passo A - Criação do Cenário de Testes no Ambiente FIBRE

Utilizamos o NOC para criar o nosso cenário utilizando duas ilhas, RNP-Brasília e UFPE-Recife, onde adicionamos os agregados de virtualização de dispositivos e redes (*OpenFlow*). O motivo de se trabalhar com duas ilhas consiste em uma forma de considerarmos uma distância geográfica entre os dispositivos de rede. Além disso, uma terceira ilha foi utilizada, a *RNP-Backbone*, sendo esta necessária para conectar as duas anteriores através da FIBREnet. Na figura 3.25 podemos visualizar a conexão das três ilhas no OCF.

Foram criadas as seguintes máquinas virtuais (*virtual machines - VMs*):

- Uma (1) VM do tipo com memória de 4 GB: *Controller*. Criada na ilha UFPE-Recife.
- Quatro (4) VMs do tipo com memória de 1 GB: *Client-xTR*, *Router1*, *Server-xTR* e *Router2-RTR*. Sendo as duas primeiras criadas na ilha UFPE-Recife e as outras duas na ilha RNP-Brasília.

Por fim, para a configuração das 4 redes (conforme mostradas na figura 3.24), foi criado um *FlowSpace* com 4 VLANs. O tráfego em cada uma dessas redes deverá ser gerenciado, obrigatoriamente, usando *OpenFlow*, pois é a única solução de virtualização de redes do FIBRE. Para tanto, devemos instalar algum controlador *OpenFlow* na rede para permitir a conectividade das máquinas virtuais. Decidimos por trabalhar com o controlador *Ryu*, cuja a instalação será descrita no Passo C, executando apenas uma aplicação

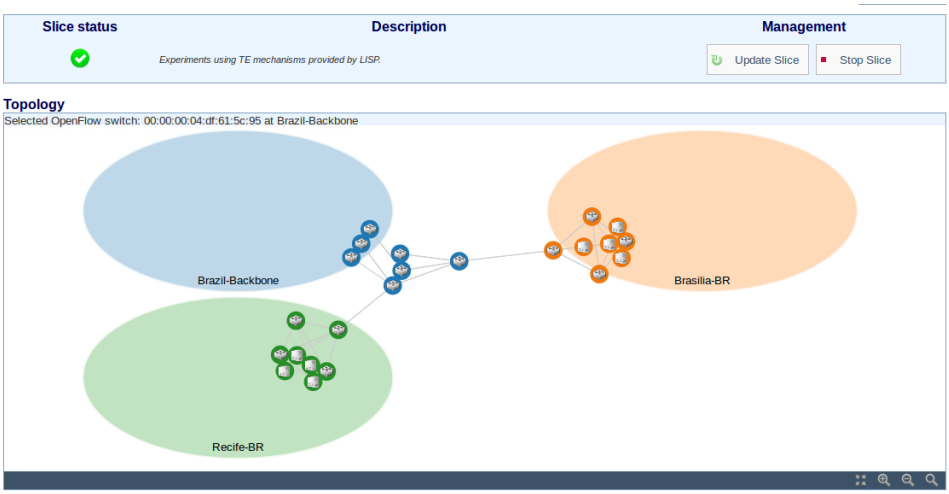


Figura 3.25. Ilhas utilizadas nos experimentos

Tabela 3.3. Dados das redes do cenário de testes

DESCRIÇÃO	ENDEREÇO DE REDE	MÁSCARA
Rede 1	192.168.1.0	255.255.255.0
Rede 2	192.168.2.0	255.255.255.0
Rede 3	192.168.3.0	255.255.255.0
Rede 4	192.168.4.0	255.255.255.0
Rede Cliente	172.18.0.0	255.255.255.0
Rede Servidor	172.19.0.0	255.255.255.0

simples de *switching*.

Passo B - Configuração das Interfaces de Rede

As interfaces de rede devem ser configuradas em todas as máquinas criadas na etapa anterior, como parte da construção de uma rede IP. Na figura 3.24 podemos visualizar a existência de 4 redes conectando os roteadores e o controlador. Além disso, de acordo com o conceito de *FlowSpaces*, introduzido pelo FIBRE, devemos atribuir cada rede a uma VLAN específica. Na etapa anterior, criamos um *FlowSpace* interligando todas as VMs e com 4 VLANs, uma para cada rede que queremos configurar.

Na tabela 3.3, podemos visualizar uma lista com as 4 redes do cenário com os seus respectivos endereço de rede e máscara. Cada rede terá disponível 254 endereços IP.

Cada VM está conectada a duas redes (duas interfaces), com exceção da VM *Controller*, que se conecta a todas as redes (quatro interfaces) com o objetivo de ter acesso a todos os dispositivos do cenário, já que é o elemento controlador.

Por padrão, nas máquina virtuais são criadas apenas duas interfaces *ethernet*: eth0 e eth1. A primeira é a interface de gerência (acesso remoto) e não deve ser modificada.

A segunda consiste na interface de dados, sendo nela onde configuraremos as interfaces virtuais. Entretanto, antes de configurá-las, devemos realizar alguns passos comuns a todos os roteadores e ao controlador:

1) Adicionar o módulo do *kernel* que o habilita a trabalhar com VLANs (*8021q*) no final do arquivo /etc/modules.

2) Editar o arquivo */etc/sysctl.conf* de forma a habilitar o IP *Forward*, para permitir o encaminhamento de pacotes; e desabilitar o *Reverse Path Filtering*, para que o *kernel* não verifique se o endereço de origem é roteável (será necessário para o LISPmob funcionar).

```
net.ipv4.conf.default.rp_filter=0
net.ipv4.conf.all.rp_filter=0
net.ipv4.ip_forward=1
net.ipv6.conf.all.forwarding=1
```

3) Agora devemos configurar a interface física *eth1* criando suas respectivas interfaces virtuais, necessárias em cada VM (*Client-xTR, Server-xTR, Router1, Router2-RTR e Controller*), editando o arquivo */etc/network/interfaces*. Para tanto, disponibilizamos as configurações necessárias para cada VM através de um pacote de arquivos[20]. As configurações originais das interfaces *lo* e *eth0* devem ser mantidas.

Para este experimento, as VLANs associadas foram 3366, 3367, 3368 e 3369. Tais valores podem mudar em outros projetos no FIBRE, entretanto basta alterá-los nas configurações acima. Além disso, como podem ser visualizadas nos arquivos baixados, rotas estáticas foram definidas nos roteadores *Client-xTR* e *Server-xTR* para que os mesmos possam se alcançar (trocar pacotes) pelas duas interfaces virtuais.

4) Reiniciar todas a VMs para que as configuração sejam efetivadas.

Passo C - Instalação e Configuração do *Ryu Framework*

O controlador *OpenFlow Ryu* foi instalado e configurado na VM *Controller* como forma de habilitar a comunicação entre todas as VMs, através do uso de uma aplicação simples de comutação na camada 2 (*switching*). O processo de instalação seguiu os passos descritos no tutorial do *site*[21] oficial da ferramenta.

A aplicação executada é a mesma executado no tutorial de instalação, a *simple_switch.py*. É importante ressaltar que selecionamos uma aplicação que trabalha com a versão 1.0 do *OpenFlow*, pois essa é a versão suportada pelos *switches* físicos do ambiente FIBRE. Ao fim desta etapa, já deve ser possível enviar pacotes de um roteador para outro.

Passo D - Instalação e Configuração do *OpenDayLight*

[20]https://www.dropbox.com/s/lrtvsstlkrdaz2z/interfaces.zip?dl=0
[21]https://github.com/osrg/ryu/wiki/OpenFlow_Tutorial

O *OpenDayLight* foi instalado e configurado na VM *Controller*, como forma de prover um plano de controle para os dispositivos de rede LISP. Como o *OpenDayLight* é uma aplicação Java, devemos inicialmente instalar o suporte a Java (versão 1.8) na máquina.

1) Adicionar os repositórios PPA da WebUdp8 para a instalação do Java via instalador da *Oracle*:

```
$ echo "deb http://ppa.launchpad.net/webupd8team/java/ubuntu \
trusty main" | tee /etc/apt/sources.list.d/webupd8team-java.list
$ echo "deb-src http://ppa.launchpad.net/webupd8team/java/ubuntu \
trusty main" | tee -a /etc/apt/sources.list.d/webupd8team-java.list
$ apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 \
--recv-keys EEA14886
```

2) Atualizar a lista de repositórios e instalação do Java 1.8:

```
$ apt-get update && apt-get install oracle-java8-installer
```

3) Criar a variável de ambiente JAVA_HOME (utilizada pelo *OpenDayLight*):

```
$ export JAVA_HOME=/usr/lib/jvm/java-8-oracle/
```

Para que essa variável seja inicializada a todo *boot* do sistema, execute o seguinte comando:

```
$ echo "JAVA\_HOME=/usr/lib/jvm/java-8-oracle/" >> /etc/environment
```

4) Fazer o *download* da versão Beryllium (0.4.0) do ODL:

```
$ wget https://nexus.opendaylight.org/content/groups/public/org \
/opendaylight/integration/distribution-karaf/0.4.0-Beryllium \
/distribution-karaf-0.4.0-Beryllium.tar.gz
```

5) Descompactar o ODL e realizar as configurações de permissões na pasta (necessárias caso queria trabalhar com o ambiente gráfico DLUX):

```
$ tar -zxvf distribution-karaf-0.4.0-Beryllium.tar.gz
$ chmod 755 -R distribution-karaf-0.4.0-Beryllium
```

6) Acessar o arquivo ***/etc/hosts*** e adicionar a seguinte linha ***"127.0.1.1 Controller"*** após a linha ***"127.0.0.1 localhost"***, para evitar erros de localização do *hostname* no Karaf.

5) Iniciar o ***Karaf***:

```
$ cd distribution-karaf-0.4.0-Beryllium
$ ./bin/karaf
```

6) Instalar a *feature odl-lispflowmapping-msmr* no *Karaf*, que inclui as principais *features* requeridas pelo serviço *LISP Flow Mapping*, como o Serviço de Mapeamento LISP e *plugin Southbound* LISP:

```
karaf> feature:install odl-lispflowmapping-msmr
```

Passo E - Instalação e Configuração do LISPmob

Instalação do LISPmob

A instalação deve ser realizada no seguintes dispositivos: *Client-xTR, Server-xTR e Router2-RTR*. Seguem os passos:

1) Para a instalação das dependências, deve-se adicionar o repositório *wheezy-backports* e, em seguida, atualizar a lista:

```
$ echo 'deb http://http.debian.net/debian wheezy-backports main' \
> /etc/apt/sources.list.d/wheezy-backports.list
$ apt-get update
```

2) Instalar as dependências:

```
$ apt-get install build-essential git-core libconfuse-dev \
gengetopt libcap2-bin libzmq3-dev libxml2-dev
```

3) Fazer o *download* do LISPmob:

```
$ git clone https://github.com/LISPmob/lispmob.git
```

4) Instalar LISPmob:

```
$ cd lispmob && make && make install
```

Configuração do LISPmob nos xTRs (*Client-xTR* e *Server-xTR*):

1) Iniciar do arquivo *lispd.conf.example*, copiando-o para a pasta */etc* com o nome *lispd.conf* e editando-o, inicialmente, removendo todas as linhas que não fazem parte da configuração do xTR.

2) Configurar o EID selecionado para *site* LISP. No nosso caso, o *Client-xTR* trabalha com o EID 172.18.0.2/32 e *Server-xTR* com o EID 172.19.0.2/32.

3) Configurar o mapeamento do EID para as duas interfaces RLOC (uma usando o parâmetro *priority* com valor 1 e outra com valor 2).

4) O endereço IP do *Map-Resolver* e do *Map-Server* devem ambos apontar para o endereço do *Controller* (192.168.1.3 para o *Client-xTR* e 192.168.2.3 para o *Server-xTR*).

5) Modificar o parâmetro *key* do *Map-Server* para uma senha de sua escolha. No nosso caso, será *"password"*.

Configuração do LISPmob no RTR (Router2-RTR):

1) Iniciar do arquivo *lispd.conf.example*, copiando-o para a pasta */etc* com o nome *lispd.conf* e editando-o, inicialmente, removendo todas as linhas que não fazem parte da configuração do RTR;

2) O endereço IP do *Map-Resolver* deve apontar para o endereço do *Controller* (192.168.3.3 ou 192.168.4.3);

3) Configurar as interfaces RTR (*rtr-ifaces*) com as duas interfaces de rede disponíveis (utilize os mesmos valores no parâmetro *priority*).

Executando o LISPmob

Para executar o LISPmob, deve-se executar o seguinte comando:

```
$ lispd -D -f /etc/lispd.conf
```

Passo F - Instalação e Configuração do *Docker*

O *Docker Engine*[22] será instalado e configurado tanto no *Client-xTR* como no *Server-xTR*, para que os *containers Client* e *Server* possam ser criados e os testes com LISP realizados. Seguem os passos:

1) Inicialmente devemos atualizar o *kernel* de *Client-xTR* e *Server-xTR*. A imagem *default* do FIBRE, utilizada para criar as VMs, consiste em uma distribuição *Debian Wheezy 64 bits* com um *kernel* linux versão 3.2. Para o *Docker* funcionar é necessário uma versão de *kernel* maior ou igual a 3.10. Portanto, realizaremos a atualização utilizando o repositório *wheezy-backports*, adicionado durante a instalação do LISPmob.

```
$ aptitude -t wheezy-backports install linux-image-amd64
```

2) Adicionar o novo *kernel* como primeiro da lista de *kernels* do arquivo */boot/grub/-menu.lst*.

```
title Debian 7 Kernel 3.16
root         (hd0,0)
kernel       /boot/vmlinuz-3.16.0-0.bpo.4-amd64 root=/dev/xvda2 ro
initrd       /boot/initrd.img-3.16.0-0.bpo.4-amd64
```

[22]https://docs.docker.com/engine/installation/linux/debian/

3) Reiniciar as VMs, para que o novo *kernel* seja utilizado. Lembrar de executar o serviço do LISPmob novamente.

4) Instalar os pacotes necessários para garantir que o APT trabalhe com HTTPS.

```
$ apt-get install apt-transport-https ca-certificates
```

5) Adicionar a nova chave GPG e adicione o repositório do *Docker* para *Debian*.

```
$ apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 \
--recv-keys 58118E89F3A912897C070ADBF76221572C52609D
$ echo "deb https://apt.dockerproject.org/repo debian-wheezy \
main" > /etc/apt/sources.list.d/docker.list
```

6) Atualizar repositórios e instalar o *Docker Engine*:

```
$ apt-get update && aptitude install docker-engine
```

7) Testar a instalação:

```
$ docker run hello-world
```

8) Caso o Passo 7 tenha sido concluído com sucesso, deve-se criar duas redes virtuais do tipo *bridge*: ***net_client*** e ***net_server***:

```
$ docker network create --driver bridge net_client
$ docker network create --driver bridge net_server
```

É necessária a criação de ambas as redes tanto no *Client-xTR* como no *Server-xTR*, como forma de forçar que a faixa reservada para net_client seja 172.18.0.0/24 e para net_server seja 172.19.0.0/24.

9) Limpar as tabelas do ***iptables***, pois o *Docker* cria regras que impedem a comunicação dos *containers* com o LISPmob.

```
$ iptables -F && iptables -t nat -F && iptables -t mangle -F
```

10) Criar e executar o *container* ***Client*** no *Client-xTR*:

```
$ docker run --net=net_client -it --name=Client ubuntu
```

Este *container* servirá como cliente para os testes com LISP, sendo executado em modo iterativo e com o IP 172.18.0.2

11) Criar e executar o *container* ***Server*** no *Server-xTR*:

```
$ docker run --net=net_server -itd --name=Server ubuntu
```

Este *container* servirá como servidor para os testes com LISP, sendo executado em modo *daemon* e com o IP 172.19.0.2.

3.5.2.2. Realização os Experimentos de Engenharia de Tráfego com LISP

A Engenharia de Tráfego deve ocorrer quando existir uma comunicação entre dispositivos clientes pertencentes a diferentes *sites* LISP. Para tanto, será realizada a criação de mapeamentos alocando diferentes rotas (RLOCs) e ELPs para pacotes destinados a um mesmo EID, seguindo critérios como prioridades e pesos.

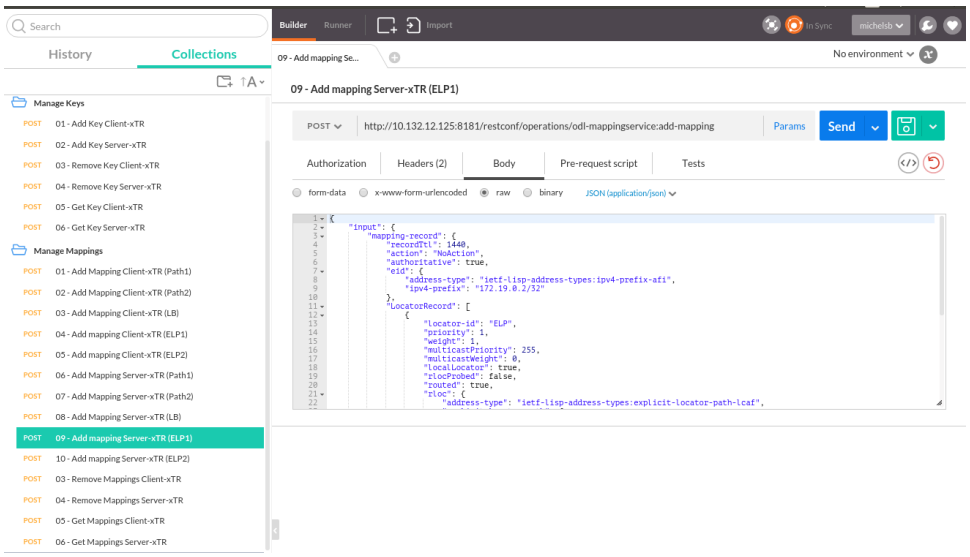


Figura 3.26. Requisições no Postman.

A variedade e característica destes mapeamentos serão definidas programaticamente no sistema de mapeamento do LISP disponibilizada pelo *OpenDayLight* (ODL), via API *Northbound* baseada em RESTCONF. Essa API permite o gerenciamento (adição, remoção ou atualização) das associações Key-EID, necessárias para o registro dos xTRs com o *Map-Server*, bem como o gerenciamento (adição, remoção ou atualização) dos mapeamentos EID-RLOC. As configurações das requisições são definidas utilizando um formato JSON.

Para a edição e o envio das requisições, foi usado o ***Postman Chrome App***[23], um cliente REST. A coleção de requisições que foram usadas nesse experimentos estão disponíveis[24], devendo ser importada no *Postman*. Para se adequar a novos experimentos, alterações devem ser feitas nas requisições modificando o endereço do *Controller* (endereço da interface eth0).

A figura 3.26 ilustra as requisições criadas após a importação. Nelas podemos encontrar exemplos que permitem a adição, remoção e consulta de associações Key-EID tanto para o EID do *container Client* (172.18.0.2/32) como para o *Server* (172.19.0.2/32), ambos com a senha "*password*". Tais associações devem, obrigatoriamente, ser criadas (executando as requisições *"01 - Add Key Client-xTR"* e *"02 - Add Key Server-xTR"*) para

[23]https://www.getpostman.com/

[24]https://www.dropbox.com/s/eng7x7hap69mis5/OLD-LISP.json.postman_collection?dl=0

que os xTRs consigam se autenticar com o *Map-Server* (lembre-se que configuramos a mesma senha no LISPmob) e consigam criar e consultar mapeamentos EID-RLOC.

Além disso, incluímos também requisições que permitem criar diferentes tipos de mapeamentos EID-RLOC para os EIDs citados acima. Tais mapeamentos permitem simular cenários de balanceamento de carga e *failover*, bem como engenharia de tráfego através da definição de ELPs.

Um exemplo de experimento com ELPs seria inicialmente executando a requisição ***"01 - Add Mapping Client-xTR (Path1)"***, que cria um mapeamento indicando que todo tráfego, destinado ao *container Client*, seja direcionado para o RLOC 192.168.101.1 (*Client-xTR*), ou seja, passando pelo *Router1*. Em seguida, deve-se executar a requisição ***"10 - Add mapping Server-xTR (ELP2)"***, que cria um mapeamento ELP indicando que todo o tráfego, destinado ao *container Server*, seja enviado inicialmente para o Router2 (192.168.102.2) e, em seguida, para o RLOC 192.168.202.1 (*Server-xTR*).

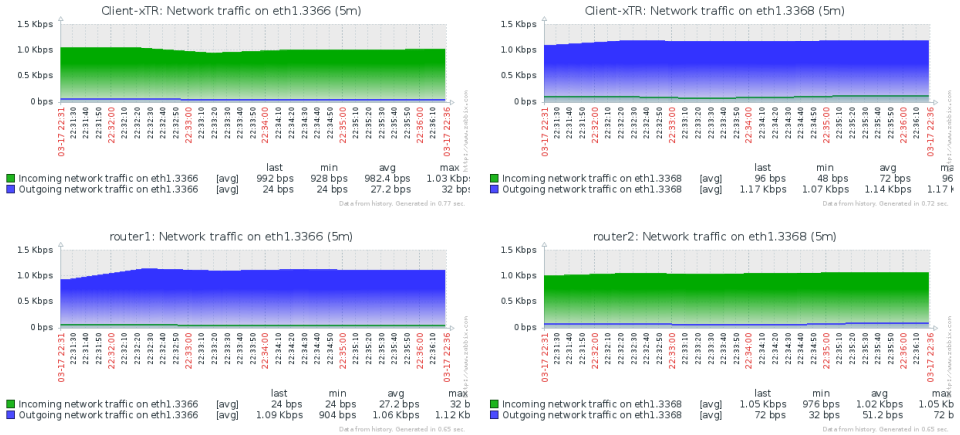


Figura 3.27. Comportamento da rede durante o ping.

Como forma de verificar o comportamento do tráfego da rede, durante os diferentes experimentos (mudanças de mapeamentos EID-RLOC), ferramentas de gerência (Zabbix[25], Nagius[26], etc) podem ser utilizadas no monitoramento do tráfego de dados em todas as interfaces de todas a máquinas virtuais. Em nossos experimentos, utilizamos o Zabbix por sua simplicidade e flexibilidade. A figura 3.27 ilustra o comportamento da rede, com a execução do exemplo de ELP anterior, durante um teste de ping realizada do *container Client* para o *Server*. Podemos perceber que o tráfego sai por uma interface (eth1.3368) do *Client-xTR* e retorna pela outra interface (eth1.3366). Podemos perceber que o tráfego sai por uma interface (eth1.3368) do *Client-xTR* e retorna pela outra interface (eth1.3366). Nesse caso, os fluxos de tráfegos de ida e volta entre os dois dispositivos finais seguem caminhos diferentes no núcleo da rede, o que pode ser utilizado para melhorar a utilização dos recursos da rede.

[25]www.zabbix.com/

[26]https://www.nagios.org/

3.6. Conclusões

Este minicurso apresentou aplicações e técnicas de Engenharia de Tráfego (ET) em Redes Definidas por *Software*, mostrando a evolução de trabalhos que utilizaram SDN para melhorar o desempenho de suas respectivas redes. Foram apresentados os principais conceitos que possibilitam a aplicação de técnicas de ET. Foi considerado como o paradigma SDN flexibiliza o desenvolvimento de mecanismos que buscam resolução para problemas de gerenciamento de tráfego nos cenários intra e inter *data center*.

O tema é amplo e há um grande potencial para pesquisa e desenvolvimento. Para tanto, problemas relevantes foram apontados neste trabalho sobre o processo de desenvolvimento de técnicas de ET em SDN, dentre eles: problemas relacionados ao desempenho e escalabilidade do controlador SDN, balanceamento de carga e alocação de largura de banda em enlaces WAN.

Além disso, experimentos foram realizados através da configuração e a execução de dois estudos de caso (EC). No primeiro EC, foram utilizados o emulador de redes *Mininet* e o controlador *OpenIRIS* como forma de utilizar o *OpenFlow* para implementar técnicas de ET. Para tanto, os testes foram realizados implementando o algoritmo *Accumulative-Load Aware Routing* (ALAR), proposto em [Nguyen and Kim 2015], que calcula o caminho mais curto considerando a carga de tráfego nos enlaces da rede. Dessa forma, foi possível melhorar a utilização dos recursos da rede, aumentar a largura de banda e reduzir o tempo de resposta. Com relação ao segundo EC, utilizamos o módulo LISP do *OpenDayLight* para implementar e explorar técnicas de ET em redes *overlay* SDN com LISP, através da criação de ELPs. Para tanto, executamos um experimento onde os fluxos de tráfegos de ida e volta entre dois dispositivos seguiam caminhos diferentes no núcleo da rede, o que pode ser utilizado também para melhorar a utilização dos recursos da rede. O cenário deste experimento foi criado e configurado sobre o ambiente de experimentação FIBRE. Ambos os processos dos experimentos foram detalhados e disponibilizados para que o público, tanto da academia como industria, possam reproduzi-los para realizarem suas próprias pesquisas sobre implementação de ET em SDN.

Referências

[Agarwal et al. 2014] Agarwal, K., Dixon, C., Rozner, E., and Carter, J. (2014). Shadow macs: Scalable label-switching for commodity ethernet. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, pages 157–162, New York, NY, USA. ACM.

[Agarwal et al. 2013] Agarwal, S., Kodialam, M., and Lakshman, T. (2013). Traffic engineering in software defined networks. In *INFOCOM, 2013 Proceedings IEEE*, pages 2211–2219.

[Akyildiz et al. 2014] Akyildiz, I. F., Lee, A., Wang, P., Luo, M., and Chou, W. (2014). A Roadmap for Traffic Engineering in SDN-OpenFlow Networks. *Computer Networks*, 71:1–30.

[Al-Fares et al. 2010] Al-Fares, M., Radhakrishnan, S., Raghavan, B., Huang, N., and Vahdat, A. (2010). Hedera: Dynamic flow scheduling for data center networks. In

Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10, pages 19–19, Berkeley, CA, USA. USENIX Association.

[Alizadeh et al. 2014] Alizadeh, M., Edsall, T., Dharmapurikar, S., Vaidyanathan, R., Chu, K., Fingerhut, A., Lam, V. T., Matus, F., Pan, R., Yadav, N., and Varghese, G. (2014). Conga: Distributed congestion-aware load balancing for datacenters. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 503–514, New York, NY, USA. ACM.

[Arregoces and Portolani 2003] Arregoces, M. and Portolani, M. (2003). *Data Center Fundamentals*. Cisco Press Fundamentals Series. Cisco.

[Barros et al. 2015] Barros, B. M., Jr., M. A. S., de Brito Carvalho, T. C. M., Rojas, M. A. T., Redígolo, F. F., Andrade, E. R., and Magri, D. R. C. (2015). Aplicando Redes Definidas por Software em Sistemas de Computação em Nuvem. *Minicursos do XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos — SBRC 2015*.

[Batista et al. 2015] Batista, D. M., Blair, G., Kon, F., Boutaba, R., Hutchison, D., Jain, R., Ramjee, R., and Rothenberg, C. E. (2015). Perspectives on software-defined networks: interviews with five leading scientists from the networking community. *Journal of Internet Services and Applications*, 6(1):1–10.

[Benson et al. 2011] Benson, T., Anand, A., Akella, A., and Zhang, M. (2011). Microte: Fine grained traffic engineering for data centers. In *Proceedings of the Seventh COnference on Emerging Networking EXperiments and Technologies*, CoNEXT '11, pages 8:1–8:12, New York, NY, USA. ACM.

[Caesar et al. 2005] Caesar, M., Caldwell, D., Feamster, N., Rexford, J., Shaikh, A., and van der Merwe, J. (2005). Design and implementation of a routing control platform. In *Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, pages 15–28, Berkeley, CA, USA. USENIX Association.

[Costa et al. 2012] Costa, L. H. M., de Amorim, M. D., Campista, M. E. M., Rubinstein, M. G., Florissi, P., and Duarte, O. C. M. (2012). *Grandes Massas de Dados na Nuvem: Desafios e Técnicas para Inovação*, chapter 1. Minicursos do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos — SBRC 2012.

[Cui et al. 2016] Cui, L., Yu, F. R., and Yan, Q. (2016). When Big Data Meets Software-Defined Networking: SDN for Big Data and Big Data for SDN. *IEEE Network*, 30(1):58–65.

[Danna et al. 2012] Danna, E., Hassidim, A., Kaplan, H., Kumar, A., Mansour, Y., Raz, D., and Segalov, M. (2012). Upward max min fairness. In *INFOCOM, 2012 Proceedings IEEE*, pages 837–845.

[Enns et al. 2015] Enns, R., Bjorklund, M., Bierman, A., and Schönwälder, J. (2015). Network Configuration Protocol (NETCONF). IETF RFC 6241.

[Farinacci et al. 2015] Farinacci, D., Fuller, V., Meyer, D., and Lewis, D. (2015). The Locator/ID Separation Protocol (LISP). IETF RFC 6830.

[FIBRE 2016] FIBRE (2016). The fibre testbed project. `https://fibre.org.br`. Accessed: 2016-03-05.

[Gharbaoui et al. 2014] Gharbaoui, M., Martini, B., Adami, D., Antichi, G., Giordano, S., and Castoldi, P. (2014). On virtualization-aware traffic engineering in openflow data centers networks. In *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pages 1–8.

[Goransson and Black 2014] Goransson, P. and Black, C. (2014). *Software Defined Networks: A Comprehensive Approach*. Elsevier.

[Greenberg et al. 2005] Greenberg, A., Hjalmtysson, G., Maltz, D. A., Myers, A., Rexford, J., Xie, G., Yan, H., Zhan, J., and Zhang, H. (2005). A Clean Slate 4D Approach to Network Control and Management. *SIGCOMM Comput. Commun. Rev.*, 35(5):41–54.

[Guo et al. 2014] Guo, Z., Su, M., Xu, Y., Duan, Z., Wang, L., Hui, S., and Chao, H. J. (2014). Improving the performance of load balancing in software-defined networks through load variance-based synchronization. *Computer Networks*, 68:95 – 109. Communications and Networking in the Cloud.

[He et al. 2015] He, K., Rozner, E., Agarwal, K., Felter, W., Carter, J., and Akella, A. (2015). Presto: Edge-based load balancing for fast datacenter networks. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 465–478, New York, NY, USA. ACM.

[Hong et al. 2013] Hong, C.-Y., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., and Wattenhofer, R. (2013). Achieving high utilization with software-driven wan. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 15–26, New York, NY, USA. ACM.

[Jain et al. 2013] Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Venkata, S., Wanderer, J., Zhou, J., Zhu, M., Zolla, J., Hölzle, U., Stuart, S., and Vahdat, A. (2013). B4: Experience with a globally-deployed software defined wan. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 3–14, New York, NY, USA. ACM.

[Jeong et al. 2015] Jeong, T., Li, J., Hyun, J., Yoo, J.-H., and Hong, J.-K. (2015). Lisp controller: A centralized lisp management system for isp networks. *International Journal of Network Management*, 25(6):507–525. cited By 0.

[Keti and Askar 2015] Keti, F. and Askar, S. (2015). Emulation of software defined networks using mininet in different simulation environments. In *Intelligent Systems, Modelling and Simulation (ISMS), 2015 6th International Conference on*, pages 205–210. IEEE.

[Kowal et al. 2015] Kowal, M., Farinacci, D., and Lahiri, P. (2015). LISP Traffic Engineering Use-Cases. Internet-Draft draft-farinacci-lisp-te-09, Internet Engineering Task Force. Work in Progress.

[Kreutz et al. 2015] Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.

[Kumar et al. 2015] Kumar, A., Jain, S., Naik, U., Raghuraman, A., Kasinadhuni, N., Zermeno, E. C., Gunn, C. S., Ai, J., Carlin, B., Amarandei-Stavila, M., Robin, M., Siganporia, A., Stuart, S., and Vahdat, A. (2015). Bwe: Flexible, hierarchical bandwidth allocation for wan distributed computing. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 1–14, New York, NY, USA. ACM.

[Lantz et al. 2010] Lantz, B., Heller, B., and McKeown, N. (2010). A network in a laptop: Rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, pages 19:1–19:6, New York, NY, USA. ACM.

[Li et al. 2014] Li, S., Shao, Y., Ma, S., Xue, N., Li, S., Hu, D., and Zhu, Z. (2014). Flexible traffic engineering: When openflow meets multi-protocol ip-forwarding. *IEEE Communications Letters*, 18(10):1699–1702.

[Liu et al. 2016] Liu, Y., Niu, D., and Li, B. (2016). Delay-optimized video traffic routing in software-defined inter-datacenter networks. *IEEE Transactions on Multimedia*, PP(99):1–1.

[McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74.

[Medhi 2010] Medhi, D. (2010). *Network routing: algorithms, protocols, and architectures*. Morgan Kaufmann.

[Moshref et al. 2014] Moshref, M., Yu, M., Govindan, R., and Vahdat, A. (2014). Dream: Dynamic resource allocation for software-defined measurement. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 419–430, Chicago, Illinois, USA. ACM.

[Nguyen and Kim 2015] Nguyen, T.-T. and Kim, D.-S. (2015). Accumulative-load aware routing in software-defined networks. In *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on*, pages 516–520.

[ODL 2016] ODL (2016). The opendaylight plataform. `http://www.opendaylight.org`. Accessed: 2016-03-05.

[ONF 2015] ONF (2015). OpenFlow Switch Specification - Version 1.3.5. `https://www.opennetworking.org/images/stories/`

downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.3.5.pdf.

[OOR 2016] OOR (2016). OpenOverlayRouter: Programmable overlays. http://www.openoverlayrouter.org/. Accessed: 2016-03-05.

[OvS 2016] OvS (2016). Open vSwitch. http://openvswitch.org/ Acessado em março de 2016.

[Rodriguez-Natal et al. 2015] Rodriguez-Natal, A., Portoles-Comeras, M., Ermagan, V., Lewis, D., Farinacci, D., Maino, F., and Cabellos-Aparicio, A. (2015). LISP: a southbound SDN protocol? *IEEE Communications Magazine*, 53(7):201–207.

[Sharafat et al. 2011] Sharafat, A. R., Das, S., Parulkar, G., and McKeown, N. (2011). Mpls-te and mpls vpns with openflow. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 452–453, New York, NY, USA. ACM.

[Sun et al. 2015] Sun, P., Vanbever, L., and Rexford, J. (2015). Scalable Programmable Inbound Traffic Engineering. In *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, SOSR '15, pages 12:1–12:7, New York, NY, USA. ACM.

[Verdi et al. 2010] Verdi, F. L., Rothenberg, C. E., Pasquini, R., and Magalhães, M. F. (2010). *Novas Arquiteturas de Data Center para Cloud Computing*, chapter 3. XXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos SBRC 2010.

[Wang et al. 2008] Wang, N., Ho, K., Pavlou, G., and Howarth, M. (2008). An overview of routing optimization for internet traffic engineering. *Communications Surveys Tutorials, IEEE*, 10(1):36–56.

[Yu et al. 2010] Yu, M., Rexford, J., Freedman, M. J., and Wang, J. (2010). Scalable flow-based networking with difane. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 351–362, New York, NY, USA. ACM.

[Zhang et al. 2014] Zhang, J., Xi, K., Luo, M., and Chao, H. (2014). Load balancing for multiple traffic matrices using SDN hybrid routing. In *High Performance Switching and Routing (HPSR), 2014 IEEE 15th International Conference on*, pages 44–49.

Capítulo 4

Revisitando Metrologia de Redes: Do Passado às Novas Tendências

Antonio A. de A. Rocha (UFF), Leobino Nascimento Sampaio (UFBA), Alex Borges Vieira (UFJF), Klaus Wehmuth (LNCC), Artur Ziviani (LNCC)

Abstract

Particularly in the last 10 years, we have witnessed an intense diversification and expansion process in the networking area concerning diferent fronts (e.g., mobile networks, virtualized networks, online social networks). In response, the area of Network Metrology followed this process by generating new proposals, characterizations, analyses, and modeling in these varied network scenarios. In this context, this short course revisits the topic of Network Metrology comprehensively and with an updated view on the applications of network measurements with respect to the network infrastructure and access, the end-user perspective, and the characterization of network traffic. We also discuss current testbeds for experimentation and validation proposals.

Resumo

Principalmente nos últimos 10 anos, houve um grande processo de diversificação e expansão da área de redes em diferentes frentes (p.ex., redes móveis, redes virtualizadas, redes sociais online). Em resposta, a área de Metrologia de Redes acompanhou esse processo gerando novas propostas, caracterizações, análises e modelagens nesses contextos variados de rede. Nesse âmbito, este minicurso revisita a temática de Metrologia de Redes de forma abrangente e com uma visão atualizada sobre aplicações de medições de rede em relação à infraestrutura e meios de acesso, perspectiva do usuário final e caracterização do tráfego da rede, além de testbeds *atuais para experimentação e validação de propostas.*

[0]Os autores estão listados em ordem alfabética do sobrenome.

4.1. Introdução e Motivação da Metrologia de Redes

Há pouco mais de duas décadas a Internet iniciou sua transformação de um instrumento restrito à comunidade científica para um componente fundamental da atual sociedade de informação. Possivelmente, a consequência mais importante do sucesso da Internet é que o propósito comum que norteava os seus componentes não mais se mantém. Atualmente, a Internet possui uma crescente comunidade de usuários, em franca expansão, que se servem de uma grande variedade de aplicações. Estas aplicações geram uma grande diversidade de tipos de tráfego e requerem novos serviços com qualidade. Em função dessa diversidade, cada componente da Internet — provedores, usuários e operadores — conscientiza-se da necessidade de melhor compreender a estrutura e o comportamento dinâmicos da rede. Entretanto, usuários, provedores comerciais de acesso, governos, operadores de telecomunicações e fornecedores de conteúdo possuem interesses que podem ser contraditórios entre si, conduzindo a uma convivência em disputa [Clark et al. 2005].

A heterogeneidade e a administração distribuída decorrentes desse cenário, aliadas à vasta cobertura geográfica e ao dinamismo, típicos da Internet em sua evolução, historicamente dificultam a caracterização da estrutura e do comportamento da rede como um todo [Floyd and Paxson 2001, Chen 2001]. Diversas abordagens baseadas em medições foram propostas para estimar e caracterizar diferentes aspectos da Internet e dos diferentes contextos de redes a ela associados, que hoje se multiplicam. O conjunto destas técnicas baseadas em medições é denominado neste texto de Metrologia de Redes.

Numa primeira fase, tipicamente no final da década de 90 e primeira metade dos anos 2000, a área de Metrologia de Redes cresceu focada em compreender a estrutura da Internet e o seu comportamento dinâmico, bem como o impacto destes no desempenho da rede [Crovella and Krishnamurthy 2006, Nucci and Papagiannnaki 2009, Wang and Loguinov 2010]. No SBRC 2005, houve um minicurso [Ziviani and Duarte 2005] que cobriu essa primeira fase de consolidação da área de Metrologia de Redes aplicada à Internet. As principais áreas tratadas nesse minicurso foram: os fundamentos de metrologia na Internet (incluindo a definição de medições ativas e passivas; bem como a métodos para a classificação e caracterização de tráfego); a estimativa de banda passante; a estimativa de matrizes de tráfego; o diagnóstico de anomalias e a estimativa de distância (em termos de atraso) na rede; as plataformas disponíveis à época para a realização de experimentos; finalizando com a apresentação de um exemplo de sistema baseado em medições de rede para geolocalização de nós na Internet. Esses eram temas em voga à época nos principais veículos publicando artigos relacionados à área de Metrologia de Redes. Diversos desses temas se mantém atuais, porém a área de medições de rede se expandiu fortemente nos últimos anos acompanhando a expansão e diversificação da área de redes como um todo.

De fato, mais recentemente, em uma segunda fase nos últimos 5 a 10 anos, houve uma ainda maior diversificação e expansão acelerada da área de redes em diferentes frentes (redes móveis, redes virtualizadas, redes sociais online, dentre outras). Em resposta a esse processo, a área de Metrologia de Redes também acompanhou esse processo de forte diversificação e expansão, gerando novas propostas, caracterizações, análises e modelagens nesses contextos variados de rede. Nesse âmbito, este minicurso revisita a temática de Metrologia de Redes (11 anos após o minicurso anterior no tema) de forma abrangente e com uma visão atualizada sobre a área.

O principal objetivo deste presente minicurso, portanto, é familiarizar novos interessados, bem como atualizar os participantes com algum conhecimento da área, com as novas tecnologias de Metrologia de Redes. Isso inclui as recentes aplicações de Metrologia de Redes em áreas emergentes e discussão das questões em aberto que podem fomentar novos trabalhos de pesquisa e desenvolvimento na área.

O restante do minicurso está estruturado como se segue. Primeiro, o minicurso oferece uma breve fundamentação teórica e histórico da área na Seção 4.2. Em seguida, o minicurso discute as aplicações de medições em diferentes contextos recentes de rede, tais como aspectos de infraestrutura (p.ex., mobilidade e virtualização) na Seção 4.3, perspectiva do usuário final (p.ex., redes de acesso e aplicações) na Seção 4.4, tráfego e roteamento na Seção 4.5 e testbeds atuais de experimentação na Seção 4.6. Finalmente, apresentamos nossas considerações finais na Seção 4.7.

4.2. Fundamentação Teórica e Visão Histórica

Nesta seção, apresentamos uma breve fundamentação teórica da área de Metrologia de Redes [Crovella and Krishnamurthy 2006, Ziviani and Duarte 2005] e uma visão histórica do desenvolvimento bem como consolidação da área ao longo dos anos.

4.2.1. Fundamentação teórica

O funcionamento básico da Internet foi concebido com o objetivo de minimizar a complexidade dos mecanismos em seu interior, concentrando o controle e a adaptação nas extremidades de transmissão. Esse princípio permitiu a expansão da Internet para as suas dimensões atuais, porém também limitou a possibilidade de monitoração do comportamento dinâmico da rede [Habib et al. 2004]. A capacidade limitada de observação da Internet é uma consequência da simplicidade do serviço de melhor esforço (*best-effort*). Tornar a rede mais observável é essencial para a verificação e a melhoria do desempenho da rede frente a aplicações mais exigentes. Além disso, uma melhor maneira de monitorar a rede se faz necessária para lidarmos com as crescentes complexidades da Internet, representada por um enorme crescimento em extensão, diversidade, velocidades de transmissão e volume de tráfego.

As abordagens baseadas em medições para a investigação de problemas relacionados com redes de computadores utilizam técnicas de medição passiva ou ativa [Barford and Sommers 2004]:

- **Medições passivas** – referem-se ao processo de monitorar o tráfego de rede sem injetar algum novo tráfego ou modificar o tráfego na rede. Isso se realiza em um ou mais pontos da rede. Medições passivas podem fornecer um conjunto detalhado de informações sobre os pontos da rede onde as medições são realizadas e sobre o tráfego de passagem por estes pontos [Arlos et al. 2005]. Exemplos são a amostragem dos cabeçalhos dos pacotes de passagem pelo ponto de monitoração ou o registro do número de perda de pacotes em um determinado intervalo de tempo. Um exemplo com pontos de medição passiva encontra-se retratado na Figura 4.1.

- **Medições ativas** – transmitem pacotes dedicados, chamados de sonda, e o resultado da travessia destes pacotes pela rede é monitorado para a inferência de carac-

terísticas da rede. Medições ativas obtêm em geral pouca informação sobre pontos isolados da rede, mas podem fornecer uma representação do caminho entre dois pontos da rede. Tipicamente, são desenvolvidas ferramentas específicas de medição ativa ou usadas ambientes de prototipação de ferramentas [Spring et al. 2003, Ziviani et al. 2012]. Na Figura 4.1, sondas de medição ativa enviadas do nó A para o nó B fornecem informação sobre o caminho que conecta estes dois pontos finais. Nesse exemplo, supõe-se extremidades na rede sincronizadas. Do ponto de vista das medições passivas, somente um dos pontos de medição registra a passagem das sondas ativas desse exemplo. Deve-se sempre considerar se o volume de tráfego introduzido por um método ativo e seu efeito no comportamento da rede estão influenciando significativamente, ou não, os resultados obtidos.

Cenários híbridos podem ser concebidos onde ambas as medições ativa e passiva são combinadas para a estimação de características da rede [Ishibashi et al. 2004].

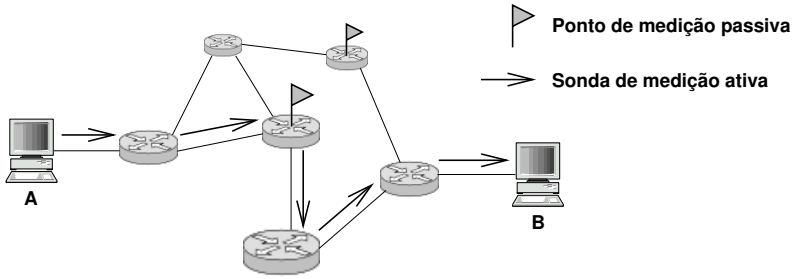


Figura 4.1. Exemplo de medições ativas e passivas [Ziviani and Duarte 2005].

4.2.2. Visão histórica

Como discutido na Seção 4.1, em uma primeira fase até a primeira metade dos anos 2000, a área de Metrologia de Redes esteve focada primordialmente na compreensão da estrutura da Internet e do seu comportamento dinâmico. Esse foco inicial procurava estudar métodos baseados em medições que influenciam diretamente outras áreas convencionais, tais como o projeto e planejamento de redes, engenharia de tráfego, qualidade de serviço e gerenciamento de redes.

Esta subseção traz uma breve visão histórica da área de medições referente a esse período anterior a 2005 através da indicação e discussão sucinta de alguns trabalhos pioneiros e relevantes. Mesmo não pretendendo ser exaustiva, essa relação de trabalhos relevantes do período certamente é representativa das atividades em voga à época na área de medições de rede. Esses trabalhos influenciaram uma série de outros trabalhos decorrentes em seu nicho de contribuição, ajudando portanto a fomentar e consolidar a área de medições e monitoramento da Internet. São eles:

- [Bolot 1993] apresentou um estudo pioneiro avaliando o desempenho da rede com uma análise simples e elegante sobre atrasos de pacotes, banda passante, tamanho de pacotes e perdas de pacotes.

- A sequência de estudos [Leland et al. 1993], [Paxson and Floyd 1995], e [Crovella and Bestavros 1997] mostraram com base em medições que o tráfego de redes locais, tráfego de redes de longa distância e tráfego WWW apresentavam a caracteristica de auto-similaridade, respectivamente. Esses foram trabalhos determinantes para se entender a natureza do tráfego de rede em diferentes perspectivas, sinalizar a necessidade de novos modelos para esse tráfego e fomentar a importância de medições de rede para estudos mais realistas.

- [Paxson 1997] investigou a dinâmica do tráfego de pacotes na Internet realizando medições de diferentes pontos e, sobretudo, mostrando como o tráfego real observado divergia dos modelos estabelecidos. Esse foi um trabalho seminal que contribuiu fundamentalmente para o estabelecimento e consolidação da área de Metrologia de Redes como metodologia de estudo do desempenho da Internet.

- Os irmãos Faloutsos mostraram em [Faloutsos et al. 1999] que a topologia da Internet apresentava propriedades de escala-livre, revelando então propriedades que impactam o desempenho da rede.

- [Spring et al. 2002] introduziram novas técnicas para o mapeamento de topologias de redes de provedores de serviço da Internet, o que permitiu a descoberta de topologias realistas com um número de medições significativamente menor.

- [Prasad et al. 2003] consolidaram nesse *survey* o conhecimento referente às ferramentas para estimativa de banda passante em suas diferentes formas: capacidade nominal, banda passante e capacidade de tranferência masiva (*bulk*) de dados.

- [Karagiannis et al. 2004] alertaram para o dinamismo de aplicações como as P2P e da consequente necessidade de desenvolvimento de novas metodologias para a medição e classificação de tráfego.

- [Lakhina et al. 2004] introduziram novos métodos para diagnosticar (ou seja, detectar, identificar e quantificar) anomalias em volume de tráfego.

- [Karagiannis et al. 2005] propuseram uma nova abordagem para classificação eficiente de fluxos de tráfego em função das aplicações que os geraram. Em contraste com métodos anteriores, essa proposta se baseava na observação e identificação de padrões do comportamento do tráfego na camada de transporte.

Há cerca de uma década, a área de redes expandiu-se enormemente tanto em infraestrutura e meios de acesso quanto em aplicações e em sua utilização por seus usuários. Tal cenário claramente aumentou a diversidade e a complexidade da Internet, o que fomentou mais desafios e demanda à área de Metrologia de Redes.

Nesse período, conferências dedicadas especificamente à área de medições na Internet se consolidaram como fóruns reputados, tais como o *ACM Internet Measurement Conference* (IMC) e os workshops *Passive and Active Measurement* (PAM) e *Traffic Monitoring and Analysis workshop* (TMA). Além dos fóruns dedicados, avanços na área de Metrologia de Redes se fazem regularmente presentes nos principais eventos técnico-científicos de redes em geral, redes móveis, gerenciamento de redes, redes sociais online,

e demais fóruns relacionados. O IETF[1] mantém grupos de trabalho dedicados a diferentes aspectos da área de medições e monitoramento de redes, tornado-se também fórum relevante de discussões relacionadas bem como fonte de informação importante no tema.

4.3. Infraestrutura

Nesta seção, tratamos aspectos de Metrologia de Redes aplicada a áreas de redes sem fio e mobilidade, caracterização de topologia em diferentes níveis e redes definidas por software.

4.3.1. Redes sem fio e mobilidade

Redes móveis e celulares estão crescendo rapidamente como principal meio de acesso de uma parcela significativa dos usuários finais. Alinhado com essa tendência, há um crecente interesse em medições do desempenho fim-a-fim em redes móveis e seu impacto em aplicações móveis. Medições desse tipo são essenciais para a adaptabilidade de serviços em redes sem fio [Sengul et al. 2012]. De forma complementar, o interesse por estudos acerca da mobilidade dos usuários também se expande recentemente uma vez que o melhor entendimento da mobilidade dos usuários gera um grande potencial de evolução e aprimoramento dos serviços oferecidos a estes usuários [Becker et al. 2013]. Nesta seção, portanto, discute-se as atuais abordagens oriundas de Metrologia de Redes para medições de desempenho em redes sem fio e móveis, bem como as tendências de estudo de mobilidade dos usuários.

4.3.1.1. Redes sem fio

A medição e monitoramento do desempenho de redes sem fio ganha importância à medida que o acesso móvel crescentemente se estabelece como meio de acesso predominante para um grande número de usuários finais. Essa crescente atividade em metrologia de redes no contexto de redes sem fio se consolida pelo interesse de diversas perspectivas no tema [Nikravesh et al. 2014]. Por exemplo, desenvolvedores avaliam o desempenho das suas aplicações móveis em ambientes com diversidade geográfica e mobilidade limitados. Pesquisadores precisam de dados de desempenho ou ferramentas para coletarem esse desempenho de modo a viabilizar experimentos realistas focados em problemas relevantes de desempenho. Operadores de rede, por sua vez, precisam monitorar o desempenho fim-a-fim de forma a manter a qualidade visada. Por fim, órgãos reguladores também têm interesse de aferir desempenho e ações dos operadores de rede. Esse cenário fomentou o surgimento de diversas iniciativas de medição de desempenho em redes sem fio, algumas das quais serão discutidas a seguir.

Nesse contexto, diversas plataformas vêm sendo propostas para a coleta de dados de desempenho em redes sem fio, elas serão tratadas mais adiante na Seção 4.6 (Validações e testbed). Um outro importante aspecto da realização de experimentos de medição é o compartilhamento de dados para a comunidade científica; e a área de redes sem fio em particular não é exceção. Haver dados de medição publicamente disponíveis referentes a diferentes cenários é importante para a simulação e avaliação de

[1] *Internet Engineering Task Force* – `http://www.ietf.org`

novas propostas na área. O repositório mais conhecido para *datasets* relativos a medições de cenários de redes sem fio é o CRAWDAD.[2] No CRAWDAD, é possível encontrar *datasets* que cobrem uma grande diversidade de perspectivas e usos de redes sem fio. Um ponto crucial na realização de campanhas de medição de desempenho envolvendo dispositivos móveis é equilibrar dois aspectos fundamentais: representatividade e viabilidade [Campista et al. 2012]. Portanto, deve-se também considerar o ganho relativo obtido com novos dados em relação ao conhecimento já existente, uma vez que novas campanhas de medição podem ser custosas e trabalhosas, sem trazer novo conhecimento significativo.

4.3.1.2. Acesso por rede celular

Um aspecto relevante em termos de demanda por infraestrutura é a crescente adoção de dispositivos móveis, em particular *smartphones*, para o acesso móvel sem fio. Logo, uma tendência recente na área de medições em redes móveis é focar em medições do uso e do desempenho de dispositivos móveis com acesso de dados via celular. Em [Fukuda et al. 2015], é analisada a evolução do uso de *smartphones*. Esse artigo mostra o uso desses dispositivos em uma gama diversa de redes: acesso via celular (3G ou LTE), acesso WiFi (2.4 a 5GHz) e pontos públicos de acesso WiFi; enquanto os usuários fazem uso de um conjunto diverso de aplicações, tais como video, sincronização de arquivos e atualizações de aplicativos. Alguns estudos buscam caracterizar o tráfego gerado e o uso realizado pelos usuários de *smartphones*. Por um lado, [Wang et al. 2015] caracterizam padrões de tráfego móvel de torres de celular de larga escala em ambientes urbanos. Por outro lado, o estudo de [Naboulsi et al. 2014] se debruça sobre a categorização de usuários pelo perfil de suas chamadas móveis e pela classificação de seus usos das redes de dados.

O advento de dispositivos móveis, como *smartphones*, com diferentes interfaces de conexão de dados, tais como WiFi ou celular 3G/4G, também abre perpectivas de melhoria de desempenho pelo uso simultâneo dessas interfaces. Estudos recentes [Chen et al. 2013, Deng et al. 2014] avaliam por medições de rede o ganho de desempenho com a adoção de conexões TCP multicaminhos (*Multipath TCP – MPTCP*) por WiFi e acesso celular simultaneamente em comparação como acesso por meio singular.

Diversos trabalhos que fazem uso de medições dedicam-se a caracterizar como os usuários estão utilizando seus dispositivos móveis, em particular *smartphones*. Por exemplo, [Chen et al. 2014] estudam como o reconhecimento do sistema operacional em uso pelos *smartphones* pode ser usado para detectar a utilização do dispositivo móvel como um meio de acesso local a múltiplos dispositivos para obtenção de uma conexão Internet. Em [Yu et al. 2014], os autores avaliam a qualidade de experiência (QoE) obtida pelos usuários de aplicações de videotelefonia, que estão cada vez mais populares. Outra aplicação bastante popular nos atuais *smartphones*, a troca instantânea de mensagens, foi caracterizada recentemente em [Zhang et al. 2015]. Outro aspecto relevante é a monitoração de segurança no contexto de aplicativos em *smartphones*. Nesse sentido, [Raghuramu et al. 2015] caracterizam o tráfego malicioso gerado por dispositivos móveis. Os autores apresentam uma discussão detalhada sobre as principais famílias de

[2] `http://www.crawdad.org`

ameaças tendo por alvo dispositivos móveis observadas no *dataset* considerado. Também são discutidas caraceterísticas estatísticas de rede que podem usadas para distinguir entre classes de tráfego legítimo e ilegítimo, o que pode ajudar na detecção de ações maliciosas em curso.

4.3.1.3. Serviço de dados de operadoras de celular

Uma tendência recente na área de medições de redes sem fio é a monitoração e aferição do serviço oferecido pelos provedores do serviço de banda larga móvel nas conexões celulares [Xu et al. 2014]. Em [Bischof et al. 2014], os autores avaliam o impacto mútuo das características do serviço (tais como capacidade, latência e perdas), a precificação do serviço e a demanda dos usuários. No Brasil, uma iniciativa relevante capitaneada pelo NIC.br é o SIMET (Sistema de Medição de Tráfego Internet),[3] que permite aos usuários aferirem a qualidade de suas conexões Internet. De particular interesse ao tema deste minicurso, é a versão SIMET Mobile, um aplicativo que pode ser instalado em *smartphones* para que os usuários desses dispositivos possam medir a qualidade oferecida por seus provedores de acesso de dados móvel. A adoção pelos usuários dos diferentes aplicativos que compõem o SIMET contribui para a formação de um mapa da qualidade do acesso Internet oferecido no Brasil.[4]

Ainda relacionado a operadoras de celular, um aspecto crucial é a avaliação do provedor mais adequado a certos perfis de usuários [Madruga et al. 2015]. Em [Joe-Wong et al. 2015], os autores se dedicam a avaliar como os planos de dados oferecidos pelos provedores afetam seu uso pelos usuários de *smartphones*. Em contraste, [Mendoza et al. 2015] estudam a composição complexa das páginas web modernas, considerando as perpectivas de acesso de estações fixas ou dispositivos móveis, de forma a identificar as principais fontes de desperdício de banda passante, o que diretamente afeta usuários de acesso móvel com planos de dados mais restritos. Por sua vez, [Kakhki et al. 2015] apresentam um aplicativo móvel para identificar se o provedor de acesso móvel suaviza o tráfego de algumas aplicações arbitrariamente, violando conceitos como a neutralidade da rede. Por outro lado, [Baltrunas et al. 2014] apresentam um arcabouço para a medição da confiabilidade de redes móveis de banga-larga. Embora uma primeira conclusão do estudo seja que a confiabilidade observada é menor que a esperada, um achado interessante mostra que, na maior parte dos casos, os dispositvos podem atingir 99,999% ("cinco noves") de disponibilidade para a conectividade combinando o serviço de apenas dois provedores.

4.3.1.4. Estudos de mobilidade

A análise de *datasets* de telefonia móvel permite a investigação da dinâmica urbana e do comportamento dos usuários em uma grande escala e nível de detalhes pouco usuais [Eagle et al. 2009, Calabrese et al. 2014]. Isso representa uma enorme oportunidade para a pesquisa, o desenvolvimento de aplicações relevantes, o diagnóstico de problemas e o planejamento futuro [Toole et al. 2015]. Em particular, a análise de chamadas telefônicas

[3] `https://simet.nic.br/`
[4] `https://simet.nic.br/mapas-app.html`

permite o estudo da mobilidade humana em uma escala sem precedentes com custo relativamente baixo [Becker et al. 2013]. Um desafio, nesse caso, é a validação dos modelos baseados em dados. A introdução de rastros de medições de referência *banchmarking* para validação é uma opção para mitigar a falta de validação explícita do modelo e tornar os modelos comparáveis [Hess et al. 2015],

Muitos estudos se dedicaram a caracterizar a mobilidade de usuários. Por exemplo, [González et al. 2008] realizaram o primeiro grande estudo de mobilidade urbana tendo por base dados de chamadas de celular, envolvendo dezenas de milhares de usuários durante um período de seis meses. O trabalho mostra uma alta regularidade temporal e espacial no deslocamento dos usuários, demonstrando a rotina da maioria dos usuários. Em contraste com o estudo do comportamento rotineiro dos usuários, outros trabalhos mais recentes se dedicaram a estudar a mobilidade dos usuários em decorrência de eventos de larga-escala em uma cidade, o que resulta em também uma demanda dinâmica, e não usual, à capacidade do provedor de acesso [Xavier et al. 2012, Xavier et al. 2013].

Mais recentemente, a previsão de mobilidade com base em dados de telefonia celular tem despertado o interesse de diferentes grupos de pesquisa [Ponieman et al. 2013]. Em [Dong et al. 2013], os autores propõe uma nova metodologia, chamada *Leap Graph*, para uma predição de mobilidade de usuários mais eficaz. A combinação de fontes de dados heterogêneas, tais como dados de chamadas de celular e informações de trânsito, é investigada em [Zhang et al. 2014]. Outra combinação de fontes heterogêneas de dados, agora usando dados de redes sociais online para complementar a base de telefonia celular, apresenta também potencial de oferecer uma previsão de mobilidade mais eficaz [Noulas et al. 2013, Silveira et al. 2015, Silveira et al. 2016].

4.3.2. Características de topologia

A caracterização de topologias sempre foi um dos grandes focos das pesquisas de medições da Internet, que buscou ao longo dos anos a implementação e aperfeiçoamento de técnicas de descoberta de topologias em diferentes níveis de conectividade na Internet: da rede, das aplicações e dos usuários. A Figura 4.2 ilustra essa estratificação da Internet atual em níveis de roteamento de pacotes na Fig. 4.2(a); de conexão entre aplicações como em uma rede de distribuição de conteúdo ou de compartilhamento de arquivos em P2P na Fig. 4.2(b); e da interação entre usuários com em redes sociais online na Fig. 4.2(c). A descoberta de topologias em todos esses níveis é bastante útil no estudo de novos protocolos e serviços e análise das infraestruturas existentes. Nesta subseção, vamos apresentar trabalhos referentes à caracterização de topologias nesses diferents níveis bem como o impacto de eventuais interdependências entre essas camadas.

4.3.2.1. Topologia de rede

O conhecimento da topologia de rede da Internet é importante para a pesquisa e o desenvolvimento de novas abordagens para a rede. No entanto, em geral, as topologias reais que compõem a Internet não são de domínio público, pois os provedores de acesso e administradores de Sistemas Autônomos (SAs) vêem as suas topologias como uma informação confidencial. Alguns provedores de acesso publicam versões simplificadas de suas topo-

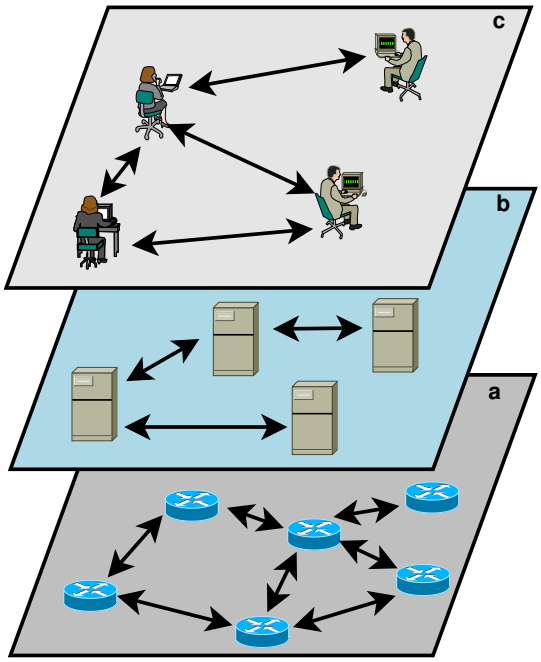


Figura 4.2. Visão da Internet atual estratificada em topologias de (a) rede; (b) aplicações; e (c) usuários.

logias, mas que não revelam detalhes ou estão desatualizadas.

A topologia da Internet pode ser representada ao nível de SAs e de roteadores. Para a inferência da topologia da Internet no nível de SAs, tradiconalmente usa-se dados das tabelas de roteamento BGP (*Border Gateway Protocol*) ou mesmo medições baseadas em `traceroute` [Zhang et al. 2005, Mahadevan et al. 2006], e variantes desta ferramenta [Augustin et al. 2006, Tozal and Sarac 2010]. Ferramentas como `traceroute` e suas variantes trazem mais informações do que a simples utilização de dados de tabelas de roteamento BGP. Isso permite utilizá-las também na inferência da topologia da Internet no nível de roteamento. Assim, ferramenta `traceroute` é adotada para a identificação dos roteadores em vários caminhos a partir de diversos pontos de medição. Alternativas para a inferência de topologia ao nível de roteamento são as ferramentas Rocketfuel [Spring et al. 2002] ou DoubleTree [Donnet et al. 2005], que buscam a inferência de topologias com mais qualidade e usando menos medições.

Mais recentemente, novos métodos baseados em busca em profundidade (*Depth-First Search* – DFS) permitiram a descoberta da topologia lógica da Internet ao nível de roteadores de maneira mais acurada e com o envio de menos pacotes sonda que as técnicas anteriores [Eriksson et al. 2012]. Um *survey* recente em [Motamedi et al. 2015] cobre de forma abrangente as propostas relacionadas à descoberta da topologia da Internet nos últimos 15-20 anos, do nível de roteadores ao nível de SAs.

Dada uma topologia de rede em um domínio, uma matriz que forneça os volumes de tráfego entre a origem e o destino em um domínio de rede possui uma grande utilidade potencial para o planejamento de capacidade e o gerenciamento de uma rede IP. O conhecimento da matriz de tráfego permite uma análise de confiabilidade para que em

casos de falha em algum enlace do domínio seja possível ao operador prever as novas cargas nos enlaces restantes. Apesar do problema de estimação de matrizes de tráfego ser tradicional e bastante investigado há cerca de uma década (ver, por exemplo, [Soule et al. 2005]), estudos recentes se debruçam novamente sobre o problema visando a síntese de matrizes de tráfego para pesquisadores que desejem testar novas aplicações ou protocolos de rede [Tune and Roughan 2015].

Além da área mais tradicional de descoberta da topologia da rede, uma tendência recente é a inferência do desempenho individual de elementos de rede (p.ex., enlaces) usando medições agregadas de caminhos fim-a-fim, em uma área conhecida como tomografia de rede. Por exemplo, é possível identificar enlaces de rede com taxas elevadas de erro através de sondas fim-a-fim coordenadas [Cunha et al. 2009]. Em [Ghita et al. 2013], os autores buscam entender os fatores que determinam a acurácia de técnicas de tomografia de rede. Por sua vez, [He et al. 2015] desenvolvem uma heurística para selecionar um subconjunto de caminhos visando a redução de erros de estimação se comparado com abordagens que distribuem uniformemente seus pacotes sonda de medição. Abordagens de tomografia de redes baseadas em comunicação multicast e unicast são avaliadas e comparadas em [Liu et al. 2015]. Um problema relevante nesse contexto de tomografia de redes é o posicionamento dos monitores responsáveis pelas medições. Há um compromisso entre o nível de identificabilidade dos enlaces envolvidos na topologia sob observação e a quantidade de mentores disponíveis [Ma et al. 2014]. Em [Qin et al. 2014], os autores oferecem um *survey* de como técnicas de *network coding* apresentam uma boa oportunidade para tomografia de redes, pois introduzem uma correlação dependente da topologia que pode ser explorada para a estimação de topologia de redes.

4.3.2.2. Topologia de aplicações

Acima do nível de conectividade físico oferecido pela topologia da rede, há uma topologia lógica entre as aplicações distribuídas que se conectam pela rede física. Possivelmente, a primeira noção de conexão lógica no nível aplicativo a ser estudada mais profundamente foi a conectividade entre documentos web no fim da década de 90 [Huberman and Adamic 1999, Adamic and Huberman 2000]. A formação de redes virtuais sobrepostas (*overlay*) sobre as redes físicas na realidade sempre esteve presente no contexto de redes de computadores, mesmo que não houvesse referência a isso com essa denominação. Um exemplo clássico é a rede de servidores SMTP (*Simple Mail Transfer Protocol*) para a troca de mensagens eletrônicas que em realidade formam uma topologia de *overlay* sobre a rede física [Lochin 2010].

O conceito de rede *overlay* ganhou força de fato com o advento do paradigma de redes par-a-par (*peer-to-peer* – P2P), inicialmente com aplicações voltadas, por exemplo, à distribuição de conteúdo [Lua et al. 2005] e realização de conectividade *multicast* no nível aplicativo [Yeo et al. 2004]. Uma proposta para monitoração de redes *overlay* baseada em tomografia de redes é apresentada em [Chen et al. 2003]. Em particular, em [Lua et al. 2005] é apresentado um *survey* de redes P2P de *overlay* tanto estruturadas quanto não-estruturadas com diferentes aplicações no domínio de redes de computadores e sistemas distribuídos. Um ponto crucial na formação e operação da rede *overlay* pode

ser o roteamento na mesma, uma vez que a proximidade no nível sobreposto pode diferir grandemente em relação à proximidade no nível físico. Assim, com base em medições de rede, é possível realizar a formação da rede *overlay* de maneira evitar um total descasamento entre a rede virtual e a física [Nakao et al. 2006, Leitão et al. 2012], com vistas a um melhor desempenho.

Medições referentes à topologia de aplicação são muito usadas para o entendimento de esquemas para distribuição de conteúdo. Por um lado, isso remete ao estudo da topologia de redes de distribuição de conteúdo (*Content Distribution Networks* – CDNs) [Pallis and Vakali 2006], mais formais de cunho corporativo para aproximação de conteúdo dos usuários finais de forma que estes percebam melhor desempenho no acesso a esses conteúdos. Por outro lado, isso também pode remeter ao estudo da topologia de redes P2P formadas espontaneamente por usuários interessados em compartilhamento de conteúdo de forma totalmente distribuída e colaborativa. Nesse contexto, há estudos sobre a topologia de uma primeira geração de redes P2P para compartilhamento de conteúdo de forma distribuída e colaborativa, comumente representada por redes do tipo Gnutella [Stutzbach and Rejaie 2005]. Em uma segunda geração de redes P2P para compartilhamento de conteúdo de forma distribuída e colaborativa, já com a incorporação de mecanismos de incentivo à cooperação e distribuição do conteúdo em blocos, sistemas do tipo BitTorrent passaram a ser muito estudados com base em medições de rede [Guo et al. 2005]. O desempenho de sistemas BitTorrent passou a ser muito estudado e a sua topologia exerce papel fundamental nesse desempenho [Qiu and Srikant 2004].

Sistemas de comunicação multimídia na Internet também se baseiam em redes *overlay* para sua estruturação e distribuição de conteúdo. Nesse contexto, o estudo da topologia dessas aplicações também fundamenta a investigação do desempenho das mesmas. Por exemplo, a escalabilidade do sistema de *overlay* do Skype é estudada em [Caizzone et al. 2008]. Mais recentemente, [Ghimire et al. 2015] propõem um modelo analítico para o estudo de desempenho de diferentes redes P2P em *overlay* usadas para o serviço de *lookup* em sistemas de telefonia IP, como o Skype. De forma similar, sistemas P2P de *streaming* de vídeo na Internet, bem como sua topologia para a provisão de vídeos ao vivo ou sob demanda, também são muito estudados [Hei et al. 2007, Liu et al. 2008, Traverso et al. 2015, Ferreira et al. 2016]. Devido ao grande impacto na rede por sua alta demanda de tráfego e volume de usuários, o serviço do Netflix também é alvo de estudos que buscam a melhor compreensão de sua arquitetura e serviço [Adhikari et al. 2012b].

4.3.2.3. Topologia de usuários

Redes sociais online tornaram-se um fenômeno global com um enorme impacto social e econômico atualmente [Heidemann et al. 2012]. Com isso, há um grande interesse em se estudar por medições de rede a topologia dos usuários nessas redes sociais, uma vez que essa estrutura é determinante para a sua compreensão, funcionalidade e novos serviços que se baseiem nessas informações. Por exemplo, difusão de informação em redes sociais online é um tema de grande interesse e está diretamente ligado à topologia de usuários [Guille et al. 2013].

Nesse contexto, surgiram os primeiros trabalhos focados na caracterização da to-

pologia de usuários de algumas redes sociais online [Mislove et al. 2007, Ahn et al. 2007], entre outros aspectos das mesmas. Esses trabalhos foram importantes para confirmar as propriedades básicas da topologia de usuários das principais redes sociais online estudadas, tais como propriedades de mundo pequeno e escala-livre com distribuição de grau em lei de potência. Outro aspecto importante mostrado foi a existência nessas redes de um núcleo densamente conectado de nós com alto grau, enquanto esse núcleo conecta pequenos grupos de nós de grau baixo altamente clusterizados nas bordas da rede. Entender essa estrutura topológica das redes sociais online é essencial para a difusão de informação e mesmo a influência no comportamento dos participantes [Centola 2010].

Em seguida, diferentes trabalhos se dedicaram a estudar especificamente as características de algumas redes sociais online populares, tais como Flick [Cha et al. 2009] ou Facebook [Ugander et al. 2011]. Em particular, [Ugander et al. 2011] caracterizam o grafo social de usuários ativos do Facebook, a maior rede social e que jamais foi analisada. Os resultados mostram que a rede social do Facebook como um todo é praticamente completamente conectada. Também é mostrado que a rede como um todo é bastante esparsa, mas o grafo de vizinhança dos usuários é bastante denso. Com mais de um bilhão de usuários de redes sociais online em todo o mundo, a pesquisa estudando a conectividade e interação de usuários nessas redes se mostra promissora para o entendimento do comportamento dos usuários [Jin et al. 2013].

Mesmo interações que normalmente não são percebidas diretamente formam redes sociais que hoje estão em estudo. Por exemplo, a relação entre vídeos relacionados no YouTube leva a uma navegação dos usuários por esses vídeos. Em particular, o conjunto dessas relações forma uma rede social com características de mundo pequeno que se mostrou uma contribuição fundamental para o sucesso do YouTube [Cheng et al. 2008]. Em [Wattenhofer et al. 2012], os autores estudam os aspectos de rede social do YouTube não somente pelo grafo de conteúdo relacionado como nos trabalhos anteriores, mas também pelos grafos de assinatura de canais e de comentários. Outro trabalho interessante, apresentado em [Jiang et al. 2013], indica que a maior parte das interações entre usuários em redes sociais online na verdade se dá por interações latentes, tais como ações passivas de visualização de perfis. Em particular, esse trabalho mostra que essas interações latentes são, além de prevalentes, também não recíprocas.

Finalmente, uma tendência recente tem sido voltada para a investigação da evolução dinâmica das redes sociais online bem como métodos para melhor caracterizar essas redes dada a atual escala em que se encontram. Por exemplo, [Antoniades and Dovrolis 2015] estudam o dinamismo co-evolucionário, indicando que a topologia da rede bem como o estado dos nós (ou enlaces) estão acoplados, de redes sociais online, com foco particular no Twitter. Em [Topirceanu et al. 2016], os autores introduzem uma nova abordagem para classificar redes sociais online por seus aspectos topológicos usando motifs (subgrafos cuja estrutura se mostre mais frequente que numa rede aleatória). Dada a atual escala das redes sociais online mais populares, abordagens baseadas em amostragem para a medição de caracteríticas importantes também se tornam alvo de investigação [Papagelis et al. 2013].

4.3.3. Redes Definidas por Software

O recente surgimento do paradigma de Redes Definidas por Software (em inglês, *Software Defined Networks* – SDN) [Astuto et al. 2014] reacendeu o interesse por arquiteturas de redes programáveis. Nesse paradigma, a arquitetura da rede é dividida em três camadas, sendo uma aplicação, uma de controle e a outra de rede física (ou fluxo). Por um lado, a visão unificada da rede oferecida pela camada de controle pode facilitar a aquisição de informações, análise de desempenho e detecção de anomalias na rede. Por outro lado, tanto a própria camada de controle como as aplicações precisam ser monitoradas e analisadas para que se possa obter um modelo adequado do comportamento da rede. Por essas razões que existe um crescente interesse da comunidade por medições de SDN. Em [Yassine et al. 2015], os atores apresentam um levantamento das principais tendências e desafios na área. Essa subseção objetiva apresentar brevemente o estado da arte assim como os principais desafios para monitoramento e análise de desempenho em SDN.

4.3.3.1. Medições em SDN

Em SDN, a estrutura de controle é desacoplada dos equipamentos de transferência de dados e das aplicações de rede. Esse tipo de construção tem vantagens sobre os modelos tradicionais, como por exemplo, a clara divisão de responsabilidades entre as partes integrantes do sistema e também a existência de um mecanismo de controle centralizado. Em particular, o controle centralizado simplifica a operação do sistema, uma vez que o controlador conhece completamente a topologia da rede e as características de cada dispositivo. Com isso, não há necessidade de se utilizar algoritmos distribuídos (ex: algoritmos de roteamento) para exercer as atividades de controle da rede. Por sua vez, ter a estrutura de controle da rede desacoplada das aplicações de rede dá flexibilidade ao sistema, fazendo com que seja possível utilizar um conjunto apropriado de aplicações de rede para cada rede. Dessa forma, é possível ter aplicações de redes com responsabilidades claras e distintas, como por exemplo engenharia de tráfego, segurança e outras. Nesse ambiente, as aplicações interagem com o controle da rede, determinando as ações a serem tomadas. O controlador, por sua vez, interpreta as instruções recebidas, verifica quais dispositivos de rede são afetados e os configura de maneira adequada.

A Figura 4.3 mostra a estrutura básica de uma SDN. Deve-se notar que, além das três camadas mencionadas anteriormente, também é necessário definir as interfaces entre essas camadas. Em geral, a interface entre as aplicações e a camada de controle é denominada interface (ou API) norte e são comumente implementadas como uma API Java ou uma API REST. Já a interface entre a camada de controle e a camada de rede física é conhecida como interface (ou API) sul. Existem alguns protocolos para a API sul, entretanto, o OpenFlow [Astuto et al. 2014, ONF 2015] se tornou o padrão de fato. Consequentemente, o OpenFlow será considerado como o padrão em uso no restante deste documento.

Como a arquitetura de uma rede SDN se divide em três camadas com funções específicas distintas, é necessário que sistemas de medições para esse ambiente sejam concebidos levando em conta essa arquitetura. Por um lado, a existência de um controlador que conhece a topologia da rede facilita que sejam feitas medições na camada de

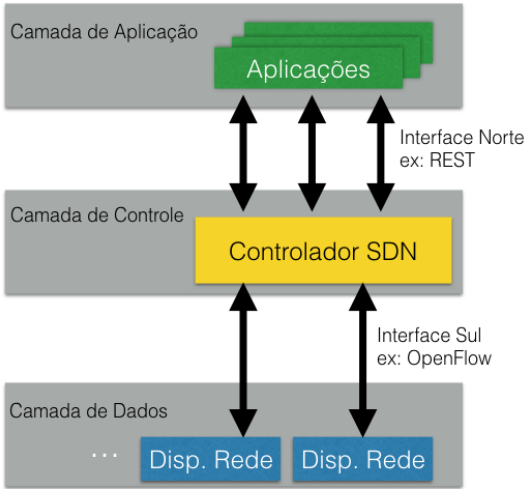


Figura 4.3. Estrutura de uma SDN.

dados. Por outro lado, a existência das camadas de controle e aplicação faz com que seja necessário realizar medições também nessas camadas, já que o comportamento das mesmas também tem impacto sobre o desempenho, segurança e usabilidade da rede como um todo. Na seqüência será apresentada uma visão de como medições são realizadas em cada uma das camadas de uma rede SDN.

4.3.3.2. Medições na camada de dados

A camada de dados é onde tradicionalmente a maior parte das medições é realizada. Em princípio, pode-se imaginar que as medições nessa camada podem ser efetuadas de maneira similar a como são realizadas em uma rede convencional. Entretanto, devido a estrutura imposta pelo padrão OpenFlow, as medidas na camada de dados são, em geral, definidas por ele, uma vez que no momento este funciona como o padrão de fato para SDN.

De acordo com o padrão OpenFlow, os equipamentos da camada de dados disponibilizam medições isoladas e agrupadas sobre os fluxos de dados que passam por eles. Essas informações podem ser coletadas pelo controlador e disponibilizadas para a camada de aplicação. Entretanto, cabe às aplicações requisitarem essas informações na periodicidade adequada para que seja possível analisá-las. Com isso, recentemente percebe-se o surgimento de vários *frameworks* construídos com o objetivo de recolher e processar as informações de medições disponibilizadas pelo controlador [Chowdhury et al. 2014, Van Adrichem et al. 2014]. O resultado disponibilizado por esses *frameworks* pode, por sua vez, ser utilizado por aplicações de QoS, engenharia de tráfego e outras que necessitem de informações derivadas das medidas disponibilizadas pelo controlador. É importante notar que, em princípio, não existe uma padronização na forma de apresentação e nem do conteúdo dessas informações. Da mesma forma, as próprias informações disponibilizadas pelo controlador evoluem conforme versões mais novas são disponibilizadas, fazendo com que o cenário de medições na camada de dados seja sujeito a constantes mudanças.

Alguns *frameworks*, como por exemplo *PayLess* [Chowdhury et al. 2014], propõem retirar da camada de aplicação a responsabilidade de controlar a periodicidade de requisição de dados de monitoramento e também oferecer medições mais acuradas e elaboradas às aplicações. Esse tipo de abordagem tem o benefício de centralizar o controle de aquisição de dados de medições do sistema e em particular de uniformizar e otimizar os aspectos dependentes de tempo do processo de amostragem e agregação das medidas disponibilizadas pelo controlador. Essa centralização também tem o benefício de garantir que aplicações distintas tenham acesso aos mesmos valores medidos, assim como otimizar o processo de aquisição dos dados evitando redundância nesse processo. Estruturalmente, esse *framework* fica localizado entre o controlador e a camada de aplicações, funcionando como um intermediário. Dessa forma, o acesso aos dados é feito através da interface norte do controlador, e por sua vez o *framework* oferece uma interface mais rica aos componentes de aplicação.

Existem também *frameworks* que visam acrescentar ao sistema ferramentas para finalidades específicas, tais como qualidade de serviços (QoS) e engenharia de tráfego. Nessa categoria, por exemplo, a ferramenta *OpenNetMon* [Van Adrichem et al. 2014] utiliza primitivas de monitoramento de fluxos, atrasos, capacidade de tráfego e perda de pacotes para facilitar o funcionamento de aplicações de engenharia de tráfego. Em particular, essa ferramenta complementa, através de medições ativas e passivas, as medições fornecidas pelo controlador SDN e, por sua vez, fornece informações adequadas para serem utilizadas em uma aplicação de gestão de tráfego. Apesar do foco em gestão de tráfego, as medições realizadas por essa ferramenta podem também ser de interesse para outras aplicações, já que oferecem informações adicionais às normalmente oferecidas pelo controlador SDN.

Uma questão fundamental nas medições do plano de dados é que estudos têm mostrado que, na prática, a qualidade das medições pode variar de acordo com os fabricantes de equipamentos. Em [Hendriks et al. 2016] os autores fazem um estudo que evidencia a variação existente na precisão das medições de contabilidade dos pacotes e bytes, assim como, a duração de fluxos de diversas implementações de OpenFlow. Assim, um dos desafios na área será o desenvolvimento de mecanismos de medição em que tais estatísticas possam ser obtidas de forma precisa e independente de fabricante.

Por fim, é preciso destacar que um importante aspecto no monitoramento de SDN é a sobrecarga que as medições do plano de dados pode levar ao plano de controle devido ao extenso número de regras nas tabelas dos equipamentos, sobretudo em redes de alta velocidade. Para lidar com essa questão, diferentes técnicas têm sido utilizadas. Em [Kreutz et al. 2014] os autores apontam que entre as principais estratégias utilizadas estão o uso de técnicas de amostragem de pacotes determinísticas e estocásticas, a estimação de matrizes de tráfego e o uso de filtros de Bloom. Essas técnicas são utilizadas para representar regras de monitoramento e fornecer medições de alta precisão, sem resultar em sobrecarga no plano de controle. Além dessas propostas, alguns trabalhos buscam separar as medições de plano de dados do plano de controle. Por exemplo, o OpenSketch [Yu et al. 2013] propõe o uso de um *pipeline* de medições de três estágios (hashing, classificação e contabilidade) para fazer o armazenamento sumarizado das informações provenientes das medições de pacotes.

4.3.3.3. Medições na camada de controle

O *Internet-draft Benchmarking Methodology for SDN Controller Performance* propõe uma metodologia para avaliação de desempenho de controladores de SDN. Por essa proposta, o controlador deve ser considerado como uma caixa preta e os protocolos utilizados nas interfaces norte e sul considerados como irrelevantes, fazendo com que a proposta seja agnóstica a esses protocolos.

As medições de desempenho propostas por esta metodologia são ativas, envolvendo a geração de uma mensagem que quando recebida pelo controlador desencadeia uma atividade que termina com o envio de uma mensagem de resposta pelo controlador. Por exemplo, a medição de tempo de provisionamento de um caminho (*path provisioning time*) é feita através da geração de um novo fluxo de tráfego. Isso faz com que uma requisição de novo fluxo seja enviada ao controlador pelo dispositivo de rede no qual o novo fluxo foi injetado, gerando uma resposta do controlador composta por mensagens para um ou mais dispositivos de rede para estabelecer o caminho pelo qual o novo fluxo será roteado. O resultado da medida é a diferença de tempo entre a mensagem recebida pelo controlador e a primeira mensagem de resposta enviada por ele. Cabe ainda notar que algumas das medições sugeridas necessita que sejam realizadas alterações de topologia da rede, como por exemplo na medição de tempo de reação do controlador a mudanças na topologia da rede. Medidas dessa natureza podem ter efeitos indesejáveis sobre o funcionamento da rede como um todo e portanto não devem ser executadas em ambiente de produção.

4.3.3.4. Medições na camada de aplicações

Na camada de aplicações residem os mecanismos de medição menos desenvolvidos. Ainda que a ocupação da interface da camada de aplicação, assim como a eficiência das próprias aplicações, possam impactar o desempenho e funcionalidade do sistema como um todo, existem poucos trabalhos que proponham medições nessa camada. Em particular, em [Isolani et al. 2015] é proposto um *framework* capaz de acessar todas as três camadas e prover uma visão integrada de medições e gestão do sistema. Entretanto, esse tipo de solução encontra dificuldades devido a falta de padronização de todas as interfaces envolvidas, assim como padronização das funções presentes na camada de aplicações.

4.3.3.5. Desafios e questões em aberto acerca de medições em SDN

Apesar das vantagens trazida pela arquitetura proposta pelas SDN, o estágio atual de desenvolvimento e padronização dessa tecnologia apresenta dificuldades para a criação de ferramentas adequadas para monitoração e gestão de desempenho dessas redes. Mesmo considerando que a arquitetura e protocolo OpenFlow sejam vistos como um padrão de fato para SDN, a padronização da interface norte e das aplicações existentes na camada de aplicação ainda se encontra em um estágio pouco definido. A volatilidade das aplicações presentes no sistema e diferenças entre implementações de controladores, em particular na interface norte fazem com que se torne difícil especificar e realizar medições nas ca-

madas de controle e aplicação, a menos de implementações considerando aplicações e controladores específicos. Entretanto, o amadurecimento dos padrões e da tecnologia em si deve, com o tempo, tornar a realização de medições nesses sistemas mais simples e padronizadas. Por fim, é preciso destacar que em cenários de SDN com virtualização, controladores distintos podem fazer uso da mesma infraestrutura física de rede operando, portanto, SDN virtuais. Assim como utilizado para sistemas operacionais, a virtualização em redes é também viabilizada através de hipervisores (do inglês, *Hypervisors*) e diferentes modelos de SDN virtuais são possíveis de implementação. Comum a todos os modelos, está a necessidade de se avaliar o desempenho das ações do plano de controle ao plano de encaminhamento, já que o uso de múltiplas SDN virtuais sobre a mesma infraestrutura física pode resultar em perda de desempenho de todas as SDN envolvidas ou na interferência de uma SDN virtual sobre a outra.

Devido às razões expostas acima, as medições são fundamentais na avaliação do desempenho de SDN. Os hipervisores, ao controlar o acesso aos recursos físicos disponíveis, podem acarretar em problemas de desempenho dos controladores, tais como o tempo de resposta que o controlador leva para responder a uma mensagem (ex. *Packet In*) e a taxa de processamento de mensagens que o mesmo pode operar na média. Por isso, alguns trabalhos focam na utilização de *benchmarkings* que apresentam referências de desempenho em ambientes de SDN não virtualizados para comparar com cenários virtualizados. Em [Blenk et al. 2016] existe uma extensa discussão sobre virtualização de SDN. Os autores apresentam diferentes modelos e arquiteturas de hipervisores, assim como, um arcabouço de avaliação de desempenho.

4.4. Usuário Final

Nesta seção, pretende-se discutir sobre as medições voltadas aos usuários finais em duas subseções distintas. São elas: (i) redes domésticas e acesso de banda larga; e (ii) medições de protocolos e aplicações.

4.4.1. Redes domésticas e acesso de banda larga

O número de usuários de banda larga fixa no mundo (inclusive no Brasil) cresceu significativamente nos últimos anos [Miniwatts Marketing Group 2016]. Só no Brasil já são mais de 24 milhões, um crescimento de 7,5% em relação ao mesmo período de 2014 [ANATEL 2015]. Da mesma forma, é cada vez mais comum aos usuários destas conexões com a Internet criarem suas próprias redes domésticas (do termo em inglês, "home networks") formadas por um número cada vez maior e dos mais diversos dispositivos eletrônicos, todos capazes de conectar a uma rede local e muitas vezes também à Internet.

Os desafios e técnicas de medição para inferência sobre redes domésticas (p.ex., número de dispositivos em operação) e aferição de métricas de desempenho (e.g., largura de banda, atrasos, perdas e disponibilidade) nas redes residenciais e nas redes de acesso de banda larga à Internet são desafios de pesquisa atuais. Esses problemas são de grande relevância para os provedores de acesso à Internet e seus usuários. Primeiro, porque esses provedores desejam conhecer o perfil de acesso dos seus clientes. Segundo, porque o problema de aferição de desempenho se tornou crucial para os governos de diversos

países do mundo, inclusive do Brasil. Tipicamente, governos passaram a exigir que os provedores de acesso cumpram com um mínimo da qualidade contratada pelo usuário. O problema é que mensurar o desempenho dessas redes não é trivial utilizando os algoritmos e ferramentas existentes e requer o desenvolvimento de novas técnicas de medição. Apresentaremos, então, os desafios e os trabalhos que vêm sendo desenvolvidos com o objetivo de monitoramento e inferência em redes residenciais e de aferição da qualidade das redes de acesso de banda larga.

4.4.1.1. Medições em redes de acesso de banda larga

No âmbito das medições de desempenho das redes de acesso de banda larga, a velocidade experimentada por muitos de seus usuários, em diversos países do mundo, especialmente no Brasil, está longe de ser apreciada. Enquanto o Brasil ocupa o *ranking* dos dez países no mundo em número absoluto de conexões de banda larga [Telecommunication Development Sector 2015], relatórios recentes da Akamai [Akamai Technologies 2014] colocam o Brasil muito aquém dessa posição em termos de qualidade, o que apenas comprova o que todo usuário já sabe: a qualidade do acesso de banda larga no Brasil deixa muito a desejar!

Segundo dados do relatório anual da Akamai [Akamai Technologies 2014], que apresenta dados referentes a 142 países, no quesito velocidade de conexão, o Brasil aparece apenas na 89^a posição. Apesar do crescimento, quando comparado a relatórios anteriores, a velocidade média no país é de irrisórios 3.0 Mbps. Abaixo da média mundial, que é de 4.5 Mbps, e uma velocidade que "ao pé da letra" nem pode ser considerada de banda larga, segundo a Federal Communications Commission (FCC) [Federal Communications Commission 2015]. Os valores registrados no Brasil são muito inferiores, não só aos dos líderes mundiais como Coreia do Sul (22 Mbps), Hong Kong (16.8 Mbps) e Japão (15.2 Mbps), mas também aos dos vizinhos na América do Sul, como Argentina (4.5 Mbps), Uruguai (5.9 Mbps) e Peru (4.0 Mbps). Os dados da Akamai ainda mostram que, os usuários brasileiros de banda larga fixa, em média, esperam mais do dobro do tempo para carregar uma página Web que os usuários do Chile e da Colômbia, e bem mais do que os usuários de todos os demais países da América do Sul considerados no relatório (Argentina, Paraguai, Uruguai e Venezuela).

Um outro fato agravante na realidade atual da banda larga no Brasil é que, uma fração significativa dos usuários experimentam velocidades de conexões (muito) abaixo daquelas informadas no contrato com as prestadoras de serviço. Relatórios mensais, produzidos pela Entidade Aferidora da Qualidade (EAQ), comprovam que os provedores estão constantemente violando as regras de garantia da Qualidade de Serviço (QoS) oferecida aos seus usuários, definidas pela Agência Nacional de Telecomunicações (Anatel) [Entidade Aferidora da Qualidade de Banda Larga 2015]. Ações semelhantes a essa estão sendo tomadas por agências regulatória e fiscalizadoras em todo o mundo, como no caso do FCC nos EUA [Federal Communications Commission 2015].

Há algumas iniciativas de pesquisadores e orgãos fiscalizadores do governo para aferição da qualidade desses serviços no Brasil e no exterior. Em [Sundaresan et al. 2011] é apresentado um estudo sobre o desempenho da banda larga nos Estados Unidos

através de monitorações feitas a partir de aparelhos da *SamKnows* (empresa especializada na avaliação de desempenho de redes de acesso e contratada pela *FCC*) instalados nas casas de mais de 4.200 usuários pelo país, além de aparelhos dos próprios pesquisadores (*BISMark*) instalados nas casas de um grupo menor de pessoas. Em 2011, a rede de pesquisa da *SamKnows* já contava com a ajuda de mais de 10.000 usuários voluntários.

Estudo similar vem sendo feito no Brasil e organizado pela *Anatel/EAQ* para medição da qualidade de banda larga. Em 2012, os organizadores do programa iniciaram o processo de recrutamento de voluntários com o objetivo de conseguirem milhares de usuários em todo o país. Os voluntários receberam em suas casas aparelhos (*white-boxes*) para medição diária e ininterrupta da qualidade da banda larga fixa fornecida por 10 grandes prestadoras do país. Dados provenientes desses aparelhos são enviados para a Anatel e periodicamente disponibilizados para consulta pública no site da agência.

Apesar dos projetos de medições de banda larga em andamento, todos esses projetos possuem um limitador fundamental de escalabilidade. Isso ocorre porque todos esses projetos dependem de uma infraestrutura e cooperação dos usuários contratantes dessas redes, pois precisam permitir a instalação de equipamentos de medições nas redes de suas residências. Com isso, o custo de um grande projeto como esse pode se tornar proibitivo para pequenas empresas ou organizações. Uma outra solução para a estimativa da largura de banda das redes de acesso residenciais seria o uso de técnicas de medições ativas e ferramentas existentes na literatura para realizar tais estimativas. Porém, [Goga and Teixeira 2012] realizam um amplo estudo em que comparam a precisão e o *overhead* das ferramentas do estado-da-arte para estimativa da largura de banda em redes de acesso residenciais. O trabalho conclui que as ferramentas falham severamente nas tentativas de estimativa, em geral subestimando a capacidade das redes por mais de 60%.

Pelo exposto, um problema que podemos considerar em aberto ainda nesta linha consiste no desenvolvimento de uma técnica não-cooperativa para aferição da qualidade de banda larga fixa através do envio de sondas de medições. Isto é, não-cooperativa que independa da colaboração do proprietário da rede de acesso, através de uma eventual intervenção na instalação, uso de alguma ferramenta ou na colocação de algum dispositivo de medição.

4.4.1.2. Monitoramento e inferências de redes domésticas

Junto com o crescimento significativo do número de residências com acesso de banda larga que vem ocorrendo em todo o mundo [Miniwatts Marketing Group 2016], cresce também o número de dispositivos, utilizados por estes usuários domésticos, com capacidade de conexão às redes residenciais, muitos deles capazes também de conectar-se à Internet. Dispositivos móveis, tais como *Smartphones*, tablets e *SmartTVs* são apenas alguns deles, e a maioria desses dispositivos são conectados através de tecnologias de redes sem fio. A quantidade e diversidade desses dispositivos utilizados pelos usuários das redes de acesso de banda larga impõe aos provedores desse serviço uma questão importante: como dar o suporte necessário aos seus clientes para os problemas enfrentados por eles em sues redes residenciais?

Embora não seja necessariamente uma responsabilidade do provedor, diagnosti-

car e resolver grande parte dos problemas enfrentados pelos seus clientes em suas redes residenciais, o usuário leigo certamente desconhece a origem do problema e até mesmo os limites das responsabilidades dos seus provedor de acesso de banda larga. Não é incomum a ocorrência de reclamação de clientes para com seus provedores de acesso de banda larga a respeito da qualidade da sua rede residencial. Estudos desmonstram, por exemplo, que um terço dos problemas reportados pelos clientes de provedores em suas chamadas de atendimento dizem respeito às suas redes residenciais [Pefkianakis et al. 2015]. Para poder dar melhor suporte aos seus usuários, provedores de banda larga e diferentes grupos de pesquisa têm tentado compreender melhor a demanda dos usuários e algumas características das redes em suas residências.

Um dos trabalhos pioneiros nessa linha foi desenvolvido por [Grover et al. 2013]. Neste trabalho é apresentado um estudo experimental de monitoramento em larga escala realizado em mais de 150 redes domésticas de 21 diferentes países do mundo, utilizando os equipamentos do projeto *BISMark* [Sundaresan et al. 2011]. Nele, foram analizados métricas relativas à disponibilidade de conexão da rede de banda larga, inferências sobre a infraestrutura da rede doméstica (incluindo a conectividade sem fio e o número de dispositivos conectados à rede), e o perfil de utilização dos usuários dasta rede doméstica. O trabalho apresenta algumas conclusões interessantes, tais como: (i) sobre a disponibilidade, países em desenvolvimento como o Brasil possuem valores muito piores nesta métrica quando comparados a países desenvolvidos. Porém, as razões não são exclusivamente por causa da má qualidade das suas conexões, mas também são atribuídas ao mau uso da rede, como por exemplo, frequentes desligamentos dos equipamentos de conexão de banda larga; (ii) sobre a infra-estrutura da rede doméstica, eles concluem que o espectro de 2,4 GHz é significativamente mais cheio do que o espectro de 5 GHz, tanto em termos de número de dispositivos quanto em termos do número de pontos de acesso visíveis, tornando assim os equipamentos configurados para esse espectro mais sucetíveis a interferência; e, (iii) sobre o perfil dos usuários, o trabalho conclui que a maior parte do tráfego de redes residenciais é destinado para apenas alguns poucos domínios, e que, em média, a maior parte do tráfego é originado de alguns poucos dispositivos, apesar da número grande de dispositivos conectados à rede. Similar ao trabalho de [Grover et al. 2013], [Sundaresan et al. 2015] apresentam um estudo que investiga como as redes sem fio afetam o desempenho experimentado pelos usuários de redes de acesso residenciais de banda larga.

Para ajudar aos provedores na mitigação e oferta de suporte aos seus clientes, [Calvert et al. 2011] defendem a criação de uma plataforma autônoma, abrangente e escalável para registro ocorrências. Segundo os autores, a criação de uma plataforma como esta é um componente chave para muitos serviços de potencial interesse aos usuários de redes de banda larga doméstica, incluindo alguns serviços que podem ajudar a mitigar os problemas de usabilidade da rede. Em [DiCioccio et al. 2013], os autores apresentam o HomeNet Profiler: um software executável em diferentes máquinas ou dispositivos, dentro de uma rede doméstica, e capaz de caracterizar métricas e coletar medidas de desempenho da rede, incluindo o número de dispositivos conectados a ela e dos tipos de serviços disponíveis (com UPnP e Zeroconf). Já em [DiCioccio et al. 2012], os autores exploram a usabilidade do serviço UPnP para medir e caracterizar as redes domésticas. Porém, os resultados deixam bastante a desejar, pois as medidas obtidas a partir do serviço

UPnP se mostram frequentemente imprecisas ou mesmo erradas.

Mais recentemente, [Sundaresan et al. 2016] apresentaram um estudo que busca responder à seguinte questão: onde há mais ocorrências de gargalos nas taxas de transferência para usuários de redes de aceeso: em suas redes domésticas ou em suas conexões com os provedores de acesso? As conclusões do trabalho sugerem que, quando a taxa de transferência do acesso de banda larga de um usuário excede cerca de 20 Mbps, uma grande fração de gargalos de rendimento são causados pela rede doméstica sem fio do usuário e não pelo canal de acesso dos provedores. Logo, nesse caso, valeria a pena buscar melhorar o desempenho da rede sem fio em casa, ao invés de tentar otimizar o desempenho em outras partes da rede de acesso e das máquinas dos usuários.

Pelo exposto, é evidente que a área de medições e monitoramente em redes domésticas e de acesso residenciais são ainda uma área com diversos problemas em aberto. A solução desses problemas são de interesse não só para os próprios usuários das redes, mas também de grande interesse para os provedores, que visam não só a satisfação dos seus clientes, mas também a redução do número de problemas reportados por seus clientes na qualidade do seu acesso à Internet.

4.4.2. Protocolos e Aplicações

Medir e monitorar o uso das aplicações e o funcionamento dos seus protocolos é importante não só para permitir uma melhor otimização dos protocolos de camadas inferiores, mas também para compreender a demanda de seus usuários e aprimorar, ainda mais, as aplicações [Xing et al. 2014]. Atualmente existem diversos relatos na literatura de iniciativas que fazem uso das medições para auxiliar o estudo, avaliação e aprimoramento de protocolos e aplicações. Nesse contexto, houve recentemente a proposta de criação de um grupo de pesquisa dedicado ao tema no IRTF: o MAPRG (*Measurement and Analysis for Protocols Research Group*).[5]

Entre os protocolos, além dos utilizados pelas aplicações tradicionais da Internet (ex. DHCP, POP3, HTTP, SMTP, DNS), é possível destacar trabalhos de medições utilizados na avaliação de redes sociais online, protocolo de aplicações multimídia, aplicações P2P e serviços de vídeo streamming. As próximas subseções apresentam as principais abordagens de medições utilizadas no suporte ao estudo e avaliação de tais protocolos e aplicações.

4.4.2.1. Redes sociais online

Redes sociais online, tais como o Facebook, Twitter, Google+, LinkedIn e Foursquare têm se consolidado como algumas das principais aplicações de usuários finais no últimos anos, sobretudo por conta do acesso a partir de dispositivos móveis [Hu et al. 2015]. Entre as redes sociais online, é preciso destacar o Facebook. Além de possuir a maior base de usuários, o Facebook se caracteriza por apresentar aos seus usuários um ambiente de convergência de serviços da Web, que resulta em um alto consumo de recursos da rede. Por tais razões, a comunidade tem dedicado esforços no sentido de não somente

[5] `https://datatracker.ietf.org/group/maprg/charter`

medir as redes sociais para entender as redes dos usuários, mas também para oferecer aos provedores de serviços de Internet um melhor entendimento dos padrões de tráfego de tais redes.

As medições de redes sociais online, contudo, impõem desafios relevantes à gerência e operação de rede devido a algumas razões. Primeiro, provedores de redes sociais online em geral não disponibilizam seus dados para a comunidade de pesquisa. Por isso, torna-se necessária a adoção de infraestruturas de medições mais complexas e mais caras, que permitam monitorar as ações dos usuários no acesso às interfaces disponíveis para que informações da rede sejam obtidas. A segunda razão é que as redes sociais online são altamente dinâmicas. A constante modificação das conexões e interações dos seus usuários dificulta a obtenção de uma visão atualizada da rede social em questão. Por conta de tais características, as abordagens utilizadas nas medições recorrem ao uso de APIs nativas da rede social, quando disponíveis, para acessar informações dos usuários ou fazem o rastreamento direto nas interfaces disponíveis. As informações obtidas por esta abordagem auxiliam a criação de grafos sociais que revelam como os usuários estão conectados e como os mesmos interagem entre si (ver Seção 4.3.2.3). Por outro lado, demais informações, relacionadas à caracterização das atividades dos usuários, não podem ser obtidas nesta abordagem, tal como o tempo de navegação em um perfil da rede [Jin et al. 2013]. É preciso destacar que dados sobre a evolução dos padrões de tráfego entre usuários e as redes sociais online podem ser bastante úteis para a gerência da rede. Por isso, medições complementares às realizadas por APIs nativas são bastante oportunas.

Uma das alternativas ao rastreamento das redes sociais online consiste na medição de dados sobre *clickstream* (seqüências de cliques) obtidos de sessões HTTP [Benevenuto et al. 2009, Schneider et al. 2009]. O monitoramento de *clickstream* pode revelar muitas informações sobre o comportamento dos usuários nas redes sociais online [Jin et al. 2013]. Contudo, esta estratégia ainda contém algumas desvantagens. Primeiro, pode existir uma falta de informações sobre os usuários com comportamento inativo, uma vez que não serão gerados dados de *clickstream*. Segundo, somente usuários monitorados pelo provedor de acesso à Internet pode prover tais dados. Demais alternativas para monitoração consistem em analisar a localidade de interesse da rede, de forma que os dados estejam mais próximos das localidades das requisições dos usuários. Além disso, fazer a análise das características de navegação a partir da análise do tráfego Web [Wittie et al. 2010, Dunn et al. 2012]. A Figura 4.4 ilustra o cenário envolvendo as duas formas mais praticadas de monitoramento de redes sociais online.

Por fim, é preciso destacar que a prática de medições será fundamental para as novas arquiteturas e plataformas utilizadas nas novas formas de redes sociais, tais como as redes sociais móveis [Hu et al. 2015]. Nesta nova geração de redes sociais, existe uma discussão para a adoção de redes oportunistas, envolvendo a conexão entre dispositivos de usuários. Em cenários desse tipo, dados de desempenho da rede, de conexões e interações de usuários deverão ser explorados. Uma discussão mais detalhada sobre essas novas gerações de redes sociais pode ser encontrada em [Hu et al. 2015].

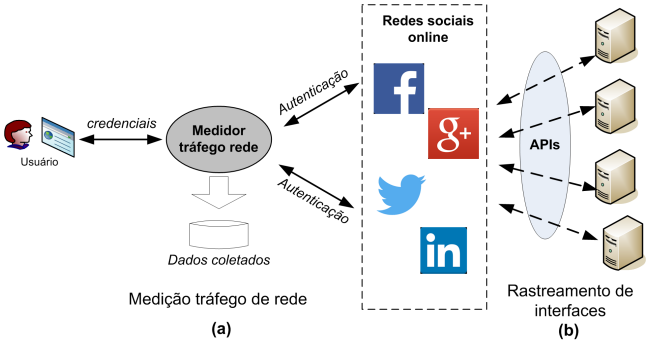


Figura 4.4. Monitoramento de redes sociais online. (a) Monitoramento feito a partir das medições de tráfego de redes. (b) Rastreamento das interfaces das redes através de APIs disponíveis (Figura adaptada de [Jin et al. 2013]).

4.4.2.2. Protocolos de aplicações multimídia: voz e vídeo

As transmissões multimídia de áudio ou de vídeo já representam a maior parte de todo o tráfego gerado na Internet nos dias atuais [Akamai Technologies 2014]. E mais, um estudo da Cisco [Cisco Systems, Inc. 2015] prevê que até 2018 o volume de dados oriundo apenas das aplicações de vídeo streaming poderá representar até 90 % de todo o trafego da Internet. Por isso, a monitoração e caracterização dessas aplicações se torna essencial nas atividades de planejamento de capacidade e engenharia de tráfego, dado o alto consumo de largura de banda.

Duas das aplicações multimídia de VoIP mais populares são Skype[6] e Google Talk/Hangout.[7] Porém, realizar medições destas aplicações não é uma tarefa trivial, uma vez que o tráfego de aplicações VoIP é de difícil identificação. O constante uso de redes sobrepostas, criptografia e mascaramento em aplicações Web, através de portas amplamente aceitas em políticas de firewall (ex. 80 e 443), inviabilizam os métodos tradicionais de monitoramento, realizados a partir do mapeamento entre tráfego e uso de portas TCP/UDP. Por tais motivos, as principais estratégias adotadas são: (i) identificação baseada em assinaturas [Adami et al. 2012, Yu et al. 2006, Yuan et al. 2014]; ou (ii) a identificação de tráfego baseada em comportamento [Freire et al. 2008, Perenyi et al. 2007, Wu et al. 2009, Bonfiglio et al. 2008, del Río et al. 2011, Gomes et al. 2013].

Na identificação por assinaturas, o tráfego VoIP é caracterizado a partir do processamento de assinaturas dos pacotes (*payloads* e cabeçalhos) [Yu et al. 2006]. O problema dessa estratégia é que as assinaturas podem não ser suficientemente únicas para a identificação e classificação do tráfego, resultando em problemas de precisão nas medições. Já a identificação baseada no comportamento consiste na análise de propriedades do fluxo Web para se inferir o tipo de tráfego escondido e de interesse de análise. No caso do VoIP, a inferência pode ser feita a partir da análise dos seguintes parâmetros: os tamanhos das requisições e respostas; o tempo de intervalo entre requisições; o número de requisições por página; e o tempo de recuperação da página. Em geral, a desvantagem dessa abordagem é a possibilidade de uma falsa caracterização do tráfego, uma vez os

[6]http://www.skype.com/
[7]https://hangouts.google.com/

valores de tais parâmetros podem ser influenciados por outras aplicações concorrentes. Na literatura é possível encontrar alguns relatos de experiências bem sucedidas com a identificação baseada no comportamento. Por exemplo, em [Freire et al. 2008] são feitos experimentos com esta abordagem para medir o tráfego de aplicações Skype e Google Talk mascarados como tráfego Web.

Mais recentemente, as estratégias de monitoração de tráfego VoIP tem sido aprimoradas, através da utilização de hardwares específicos para sua captura e análise. Por exemplo, em [Antichi et al. 2014], placas FPGA são utilizadas no desenvolvimento de uma solução de baixo custo para medição de parâmetros de QoS de tráfego VoIP em enlaces de alta capacidade.

Dado o sucesso de serviços como YouTube,[8] Netflix[9] e Hulu,[10] as redes de distribuição de conteúdo desses serviços têm sido objeto de estudo da comunidade científica e, conseqüentemente, a prática de medições é recorrente [Adhikari et al. 2015]. Os estudos e avaliações dos serviços dessas aplicações de distribuição de video sob demanda têm sido propostos através de abordagens tanto de medições ativas quando passivas em anos recentes.

Em termos de medições ativas, a metodologia geralmente adotada para medições de aplicações como YouTube consiste em desenvolver clientes do serviço, que são implantados em pontos estratégicos da rede e fazem o envio de requisições aos servidores da nuvem. Os fluxos de tráfego de resposta de cada requisição são utilizados na análise de desempenho da rede em geral ou da rede de distribuição de conteúdo utilizada para o provimento do serviço. Para a adoção dessa estratégia é preciso lembrar que o sistema de DNS da Google leva em consideração a localidade do usuário na sua resolução de nomes para IP, respondendo com o endereço IP mais próximo ao servidor de DNS do usuário requisitante do vídeo. Por isso, algumas plataformas de medições ativas para aplicações como YouTube necessitam ser geograficamente distribuídas. Nesse caso, recursos disponíveis em redes de experimentação (testbed) como o PlanetLab tornam-se essenciais. A Figura 4.5 ilustra a configuração de uma plataforma de medições deste tipo de aplicação. A figura mostra a rede do testbed utilizada para fazer requisições à infraestrutura de distribuição de conteúdo em nuvem da Google para disponibilização de vídeos do YouTube. Esta rede também pode ser utilizada para resolução de nomes através do DNS e emular players desenvolvidos em Flash para o YouTube. Além disso, também uma infraestrutura de resolução de DNS da nuvem do YouTube é necessária para identificar os resultados de traduções de nomes para servidores locais. Por fim, um cluster de máquinas utilizado no processamento e avaliação do tráfego de vídeos recuperados.

Algumas iniciativas que fazem uso das medições ativas são descritas na literatura. Em [Adhikari et al. 2012a], foi construída uma infraestrutura distribuída de medições ativas para, através de uma espécie de engenharia reversa, verificar o sistema de distribuição de vídeos do YouTube e obter respostas sobre como o serviço projeta a sua infraestrutura de serviço de vídeos, tendo como base a localização geográfica dos seus usuários. Nesse trabalho, as medições também foram utilizadas para verificar como o balanceamento de

[8]https://www.youtube.com

[9]https://www.netflix.com

[10]http://www.hulu.com (serviço restrito para uso em território americano)

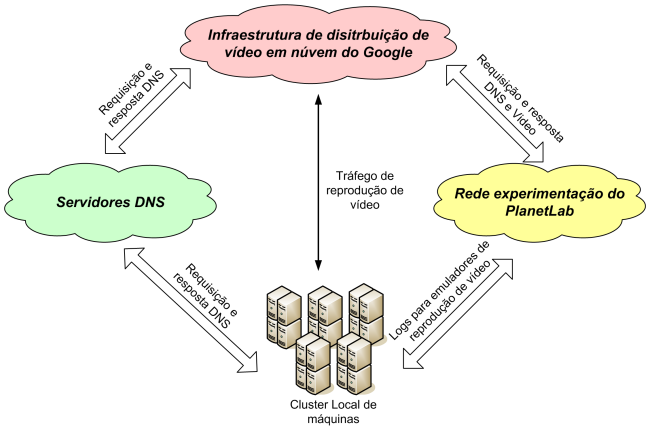


Figura 4.5. Infraestrutura distribuída de medições de desempenho de aplicações como o YouTube (Figura adaptada de [Adhikari et al. 2012a]).

carga é realizado nos servidores localizados em diferentes regiões geográficas. Uma vez que o sistema de serviço de vídeos do YouTube é amplamente distribuído através da sua rede de nuvem, a infraestrutura de medições adotada necessita ser igualmente distribuída. Para isso, o experimento apresentado em [Adhikari et al. 2012a] fez uso dos nós disponíveis no PlanetLab para fazer medições ativas distribuída na infraestrutura de nuvem do YouTube, onde cada nó do PlanetLab utilizado no experimento emula um player do serviço. Outros experimentos com o YouTube realizados através de medições ativas são feitos para identificar propriedades da rede, como por exemplo, em [Ahsan et al. 2015] onde clientes de YouTube foram utilizados para comparar o desempenho do serviço em redes IPv4 e IPv6.

Em ambientes mais controlados, o tráfego do YouTube pode ser monitorado através de medições passivas. A partir das informações da troca inicial das mensagens de HTTP, os cabeçalhos das requisições subsequentes, destinadas aos servidores, podem ser capturados para gerar estatísticas de tráfego. Em [Krishnappa et al. 2013, Zink et al. 2009] esta estratégia foi adotada através do uso de placas de captura DAG.[11] Por fim, é preciso destacar que, assim como constatado em serviço como YouTube, outros serviços de conteúdo multimídia têm sido objeto de estudo da comunidade. Os mais relevantes são os provedores de vídeo sob demanda Netflix e Hulu. A avaliação dos mesmos também já foi feita através da abordagem de medições ativas para a obtenção de métricas de desempenho, como demonstrado em [Adhikari et al. 2015].

4.4.2.3. Aplicações P2P

Um grande número de aplicações P2P são utilizadas na Internet resultando num significativo consumo de recursos da rede [Fattaholmanan and Rabiee 2015]. No passado, um extenso número de trabalhos relacionados à monitoração de aplicações P2P foram publicados. Em alguns desses trabalhos, as informações de desempenho foram utilizadas na análise do comportamento das aplicações, modelagem, análise das cargas de trabalho e

[11] http://www.endace.com/endace-dag-high-speed-packet-capture-cards.html

caracterização de tráfego [Zhou et al. 2007, Pouwelse et al. 2005, Tutschku 2004, Stutzbach and Rejaie 2006, Gummadi et al. 2003]. Num segundo momento, algumas das atividades de monitoração estiveram voltadas para o suporte ao rastreamento de conteúdos protegidos por propriedade intelectual distribuídos ilegalmente na rede [Fattaholmanan and Rabiee 2015, Siganos et al. 2009].

Atualmente, é possível identificar iniciativas de monitoramento de aplicações P2P, como o BitTorrent[12] e SopCast,[13] que fazem uso intensivo de largura de banda para distribuição e compartilhamento de conteúdos digitais. Alguns trabalhos de monitoração de aplicações P2P têm sido direcionados para a caracterização de fluxos de tráfego *online* de conteúdo digital de sistemas de transmissão de televisão e rádio [Ciullo et al. 2010, Fallica et al. 2008, Silverston et al. 2009], tais como SopCast, PPLive,[14] e UUSee.[15] Algumas dessas iniciativas fazem uso das informações dos nós a fim de identificar diversas propriedades da rede P2P. Por exemplo, no experimento descrito em [Ferreira et al. 2016] foram coletados dados do SopCast a partir de clientes do serviço instalados em máquinas do PlanetLab para implementar uma estratégia de predição do nível de cooperação dos nós na rede overlay. Em outros trabalhos relacionados, busca-se a descoberta de topologia, a partir de informações obtidas nas mensagens de controle, enviadas pelos nós (*peers*) para entidades centralizadoras (*trackers*). Essa informação é então utilizada na avaliação do nível de cooperação entre os nós. Por fim, [da Silva and Rocha 2015] apresenta uma arquitetura de monitoramento da rede BitTorrent para determinar e classificar os *peers* dentro de um enxame, onde foram detectados a existência de uma quantidade significativa de usuários tirando proveito de uma rede de compartilhamento de arquivo para realizar *streaming* de vídeo.

Em termos de desafios e tendências no monitoramento de redes P2P é possível afirmar que o monitoramento motivado pelas questões relacionadas à propriedade intelectual dos conteúdos disponibilizados na rede continuará em voga, sobretudo trabalhos que investiguem os ataques às redes P2P como forma de inibir a disponibilização ilegal de conteúdos digitais [Fattaholmanan and Rabiee 2015]. Além disso, as pesquisas que investigam as características de topologias de redes P2P recorrerão às medições para obter, por exemplo, dados de popularidade dos nós [Gunduz and Yuksel 2016]. Por fim, medições serão fundamentais no suporte ao estudo de aplicações P2P utilizadas em cenários de redes tolerantes a atrasos (DTN – *Delay-Tolerant network*) e de computação pervasiva, devido à mobilidade, dinamicidade e heterogeneidade dos elementos de rede em tais cenários [Malatras 2015].

4.4.2.4. Serviços de armazenamento em nuvem

Os serviços de armazenamento em nuvem são uma classe de aplicações em forte expansão nos últimos anos [Zenuni et al. 2014, Drago et al. 2012]. Tais serviços são providos por aplicações mundialmente conhecidas, tais como Dropbox, Microsoft SkyDrive, Apple

[12]http://www.bittorrent.com
[13]http://www.sopcast.org
[14]http://www.pptv.com
[15]http://www.uusee.com

iCloud and Google Drive, que se propõem a manter o armazenamento de arquivos de usuários em infrastruturas de nuvens, de forma sincronizada e transparente para o usuário final. Para esse tipo de aplicação, medições passivas e ativas têm sido utilizadas para avaliar o desempenho de tais aplicações e o consumo de recursos de rede [Wang et al. 2012, Gracia-Tinedo et al. 2013, Drago et al. 2013, Lopez et al. 2014].

Em [Gracia-Tinedo et al. 2013], os autores recorrem às medições ativas para fazer a análise do uso do serviço de armazenamento em nuvem. Para isso, são criados clientes da aplicação, desenvolvido através de APIs REST disponibilizadas para desenvolvedores. Esses clientes são instalados em pontos espalhados através de redes privadas (Universidades, por exemplo) e de nós do PlanetLab, a fim de fazerem chamadas aos servidores da núvem. Outras iniciativas com o uso de medições de ativas estão descritas em [Drago et al. 2013].

As medições passivas também são utilizadas para medição deste tipo de aplicação. Em [Drago et al. 2012], o Dropbox foi analisado a partir da observação do protocolo da aplicação executado em conexões TCP. Os autores fizeram uso de uma ferramenta de medições passivas para extrair informações de certificados de conexões HTTPS utilizado no serviço. A mesma abordagem é também adotada em outros trabalhos [Wang et al. 2012, Lopez et al. 2014].

O conceito de serviços de armazenamento em nuvem, disponível através de aplicações como Dropbox, pode ser também considerado como um conceito de serviço de nuvem pessoais (*Personal Cloud*). A popularização desse tipo de serviço já vem despertando o interesse da comunidade científica em obter dados que facilitem o entendimento dessas aplicações e os impactos no desempenho da rede devido a suas utilizações [Tang et al. 2015, Gracia-Tinedo et al. 2015]. Por tais motivos, trabalhos de medições para esses tipos de aplicações começaram a aparecer mais fortemente a partir de 2014, por exemplo em conferências como o IMC (*Internet Measurement Conference*). Por isso, é possível afirmar que essa é uma das áreas em que estudos e técnicas de medições voltadas para esse tipo de aplicação será bastante útil, principalmente se considerarmos o potencial de expansão dessas aplicações para os próximos anos.

4.5. Tráfego

Pesquisadores têm desenvolvido soluções para flexibilizar os objetivos configuráveis em uma rede. Por exemplo, novas soluções flexibilizam o roteamento em redes de trânsito via comutação por segmentos ou extensão a protocolos de roteamento intradomínios. Há propostas até mesmo de linguagens para expressar mecanismos para compor políticas complexas de roteamento. Nesse cenário, tanto a manutenção de rotas, quanto o tipo de tráfego são diferentes do cotidiano. Nesta seção, iremos discutir os desafios e técnicas para realizar medição de tráfego e roteamento em cenários atuais. Focaremos, principalmente, em centros de processamento de dados que tem sua flexibilização calçada no paradigma de redes definidas por software. Apresentaremos os recentes trabalhos que tratam de medições e amostragem de tráfego nesses ambientes e desafios a serem enfrentados, como uma amostragem de dados eficiente.

4.5.1. Composição atual de aplicações e tráfego de rede

Notoriamente, a Internet cresce a cada dia, tanto em abrangência quanto em volume de tráfego. Acompanhando tal crescimento, mais e mais redes, em diferentes partes do mundo, contam com esse meio para a troca de informações. Mais ainda, além desse crescimento, observamos novas aplicações que mudam o modo de usar a Internet. Como resultado desse cenário dinâmico, surge uma pergunta simples, porém ainda sem resposta definitiva: "*como é a composição de aplicações e tráfego na Internet de hoje?*" [Richter et al. 2015]. Claramente, ter respostas para essa pergunta é importante para tarefas como identificar o surgimento de novas tendencias de uso da Internet, otimizar o desempenho de aplicações e alocar recursos na rede de forma eficiente.

Há várias maneiras de se classificar o tráfego observado na Internet e, assim, determinar o conjunto de aplicações. A seguir, descrevemos brevemente as mais populares:

- **Abordagem baseada em porta:** muitas aplicações executam em um número de porta fixo (ou algumas portas bem determinadas). Assim, a tarefa de classificar tais aplicações pode ser realizada pela captura de pacotes e simples análise de seus cabeçalhos.

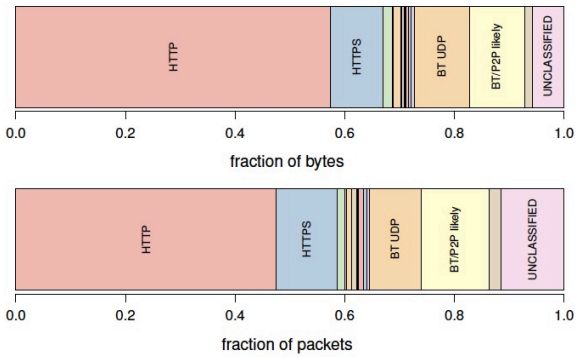
- **Abordagem baseada no conteúdo (payload):** essa abordagem é conhecida como *Deep Packet Inspection (DPI)*. Nela ela, a classificação é realizada por assinaturas específicas, encontradas no conteúdo do pacote de dados. Por exemplo, pode ser utilizado sequências de bytes de protocolos bem conhecidos. Tipicamente, essa abordagem é baseada nas mensagens de estabelecimento de conexão do protocolo (*handshake*) e, assim, só necessitam avaliar alguns poucos pacotes de dados que carregam essas mensagens iniciais.

- **Abordagem baseada nas características do fluxo:** essa abordagem utiliza outras propriedades do fluxo, além da porta de origem-destino. Por exemplo, pode ser utilizado o total de pacotes do fluxo e seu tamanho médio. Há diversas ferramentas que tem sucesso na classificação de pacotes ao se utilizar essa abordagem, sem a necessidade derealizar inspeções no conteúdo dos dados. Porém, em casos de fluxos muito grandes, a classificação pode ser custosa com relação à memória.

- **Abordagem baseada no comportamento do hospedeiro:** classificadores que utilizam essa abordagem verificam o perfil de interação entre os hospedeiros envolvidos. Eles podem ver quais hospedeiros são contactados, quais portas são utilizadas e a periodicidade de tais contatos. As diversas abordagens dessa classe são particularmente efetivas na classificação de tráfegos de aplicações P2P.

4.5.1.1. Composição atual de aplicações e tráfego da Internet

Geralmente, não é adotada uma única abordagem para classificação de tráfego. Por exemplo, [Richter et al. 2015] desenvolveram uma metodologia para classificar o conjunto de aplicações a partir de uma amostra de pacotes, coletados em um dos maiores pontos de troca de tráfego na Europa. O método desses autores utiliza uma combinação de várias



Protocol	% bytes	% pkts
HTTP	57.39	47.52
HTTPS	9.53	11.08
RTMP	1.72	1.48
NNTP	1.41	0.87
NNTPS	0.63	0.38
SMTP	0.53	0.91
DNS	0.45	0.87
SSH	0.42	0.61
other known	0.68	0.74
BT UDP	10.00	9.57
BT/P2P likely	10.14	12.31
P2P likely	1.32	2.10
unclassified	5.78	11.56

Figura 4.6. Composição de aplicações para pacotes e bytes em setembro de 2013 [Richter et al. 2015].

abordagens, tais como: verificação de assinatura nos dados, padrões de comunicação dos hospedeiros e classificação baseada em porta. Nesse caso, os autores mostram que o método proposto consegue classificar cerca de 95% do tráfego.

A Figura 4.6, apresentada em [Richter et al. 2015], mostra claramente que o tráfego HTTP(S) domina o conjunto de aplicações, com uma porção de mais de do que 65% dos bytes classificados. Enquanto há um crescente domínio de HTTP para uma infinidade de aplicações – já relatado em vários estudos anteriores–, também há uma parte significativa do tráfego composta pelo BitTorrent (UDP) e uma provável classe BT/P2P, representando cerca de 20% dos bytes observados. Outros protocolos, como e-mail, newsgroups, RTMP, etc. contam com apenas cerca de 6% dos bytes observados no ponto de troca de dados.

Ainda de acordo com [Richter et al. 2015], HTTP(S) domina a composição de tráfego de aplicações ao longo de um período, nunca sendo menor que 55% do tráfego observado. Mais ainda, observa-se uma característica diuturna, indicando um uso pronunciado de aplicações com tráfego HTTP(S) nas horas de maior movimento do dia, e fim de tarde. Os picos de contribuições de Bittorrent e P2P são fora do horário comercial. Curiosamente, observa-se um segundo pico de atividade BT/P2P em cada dia, que provavelmente ocorre por conta dos usuários dessas aplicações estarem em diferentes fusos horários.

4.5.1.2. Composição de aplicações e tráfego em redes domésticas e sem fio

O conjunto de tráfego e aplicações da Internet pode se diferenciar de forma significativa do encontrado em redes domésticas. Nessas redes, há o predomínio de comunicação sem fio através do protocolo de rede sem fio IEEE 802.11. De fato, o baixo preço, associado aos dispositivos móveis (como celulares), tornam a adoção de WiFi muito atrativa.

Mesmo com tal popularidade, ainda se conhece pouco do desempenho e do tráfego de tais redes. Para transpor essa falta de conhecimento, [Sundaresan et al. 2015] realizam medições passivas em pontos de acessos sem fio de usuários domésticos. Os auto-

res criaram uma ferramenta que executa em pontos de acesso facilmente encontrados no mercado, o que facilita a execução dos experimentos e o espalhamento da ferramenta de avaliação. Tal ferramenta coloca traços de pacotes de conexões que passam pela interface *Wan* e de ambas as interfaces sem fio existente nos pontos de acesso testados. Para tal tarefa, os pesquisadores utilizam o *pcap*, uma ferramenta amplamente conhecida e utilizada para se capturar dados em interfaces de rede. Nesse trabalho, os autores mostram que o tráfego de um usuário em uma rede desse tipo raramente atinge a taxa de transferência total do enlace. Eles apontam que a própria demanda de tráfego do usuário pode ser insuficiente (estudos anteriores mostraram que este fenômeno é frequente [Siekkinen et al. 2007]). Outra razão apontada é relacionada ao tamanho dos fluxos: como requisições web são predominantes, estas são pequenas (e curtas) demais, impossibilitando que estes fluxos saturem o enlace.

A latência dentro de uma rede doméstica é, em muitos casos, o principal fator de impacto no tempo fim-a-fim de ida e volta de um pacote (RTT). Nesse caso, as redes sem fio contribuem com uma latência média de cerca de 8 ms. Entretanto, cerca de 30% do dispositivos avaliados experimentam um RTT local maior que 15 ms [Sundaresan et al. 2015]. Os autores, naquela época, notaram que comunicação simultânea ocorre com pouca frequência. A maioria dos locais que eles implantaram seus equipamentos de medição tinha vários dispositivos associados. No entanto, curiosamente, esses dispositivos muitas vezes não eram muito ativos, especialmente ao mesmo tempo que outros. Os autores realizaram medições a cada segundo, observado o número de vezes que vários dispositivos enviaram, pelo menos, 25 pacotes dentro do intervalo de medição. De acordo com seus resultados, para 85 % dos intervalos de um segundo na banda de 2,4 GHz e 93% dos intervalos na banda de 5 GHz, observou-se, no máximo, um dispositivo enviando dados.

4.5.1.3. Geração de matrizes de tráfego

Tão importante quanto a composição de aplicações e tráfego de uma rede, é a matriz de tráfego que encontramos nela. A falta de dados públicos acerca de matrizes de tráfego continua sendo um obstáculo às pesquisas em diversas áreas (principalmente para planejamento de redes) [Tune and Roughan 2015]. Para suprir tal lacuna, pode-se utilizar cargas sintéticas. De fato, testes não (necessariamente) exigem entradas realistas, uma vez que o objetivo é entender o efeito que uma entrada específica tem em um algoritmo. Nesse sentido, [Tune and Roughan 2015] propõem uma abordagem de geração de matrizes de tráfego sintéticas que evita o paradigma medição-modelo ao aplicar o princípio de entropia máxima [Jaynes 1957]. Apesar do uso de teoria da informação e entropia em processos anteriores para geração de matrizes de tráfego, a abordagem proposta não necessita de nenhum dado medido, mas, se o tiver, esses dados podem ser incorporados para geração da matriz de tráfego.

4.5.2. Tráfego em redes de distribuição de conteúdo

Atualmente, grandes serviços web servem seu conteúdo a partir de múltiplos locais para reduzir a latência percebida pelo cliente, para distribuir a carga e para fornecer redundância contra falhas. A forma mais comum de realizar esse processo é através do uso de redes de

distribuição de conteúdo (CDN). De páginas web tradicionais a conteúdo mais elaborado como vídeo sob demanda (VoD), CDNs são o elemento chave para distribuição eficiente de conteúdo [Fan et al. 2015, Imbrenda et al. 2014]. Por exemplo, a maior parte dos grandes publicadores de conteúdo, como NetFlix, Hulu, Facebook, Pinterest e CNN, usam CDNs. Mais ainda, quase sempre tais CDNs pertencem a terceiros. Há relatos que mais de 70% de todo tráfego Internet é carregado por CDNs [Labovitz 2012].

CDNs são compostas por vários *clusters*, que por sua vez são compostos por diversos servidores. De maneira geral, tais *clusters* estão geograficamente espalhados pelo mundo. A CDN direciona dinamicamente usuários para *clusters* na granulosidade de rede mais próxima a eles. Assim, a CDN pode direcionar o usuário por regras de roteamento (e.g. *anycast* mais BGP) ou usando um algoritmo de mapeamento de DNS.

Idealmente prefixos do usuário poderia ser usado para direcioná-lo a um cluster mais próximo, e assim, minimizar a latência do serviço. Na prática, esse mapeamento é mais complicado. Tal decisão pode ser influenciada por *clusters* temporariamente indisponíveis, outros sobrecarregados, até mesmo por estimativas de proximidade incorretas ou ultrapassadas; além de questões como tarifação de tráfego entre *peerings*.

Assim, [Fan et al. 2015] relatam que usuários experimentam frequentes mudanças no mapeamento para a CDN. Cerca de 20% de prefixos do Google e 70% dos prefixos da Akamai mudam de mapeamento cerca de 60 vezes durante uma investigação de 1 mês (uma média de duas vezes ao dia). Essas mudanças afetam o desempenho percebido pelos usuários. Durante o mês de avaliação, entre 50 e 70% dos prefixos têm seus mapeamentos alterado para um cluster que é mais distante, e algumas vezes (entre 28 e 40%), essas mudanças resultam em latências maiores. Grande parte das vezes, essas mudanças são temporárias, porém, alguns poucos usuários (entre 2 e 5%), percebem um desempenho pobre por todo o tempo. Para conduzir tais medições, mapeando prefixos de IP em clusters da CDN e também avaliando o desempenho percebido pelos usuários, [Fan et al. 2015] contaram com pontos de coleta de dados no PlanetLab. Eles dispararam *ping* desses pontos de coleta aos clusters da CDN. Eles também avalianram respostas DNS do PlanetLab para os nomes de domínios investigados.

Nem todo tipo de conteúdo, entretanto, obtém benefícios pelo uso imediato de CDNs. Há uma dependência clara, assim como em qualquer cache, de sua referência temporal. As características chaves de um tráfego de um grande ISP foram monitorados em [Imbrenda et al. 2014]. Os autores avaliaram a popularidade do conteúdo e o potencial desse ser armazenável em CDNs (ou caches).

O potencial de um conteúdo ser armazenável em cache (*cacheability*) é uma métrica que, dada uma análise temporal, indica qual fração de requisições por um determinado objetio será realizado mais de uma vez em um determinado período. A primeira requisição a um objeto não é considerada cacheável, enquanto requisições subsequentes, no mesmo período de análise, são. A Figura 4.7 apresenta os principais resultados desse trabalho, com relação a conteúdo cacheável, dada a visão de um ISP. Nas figuras a-b, são apresentadas as reduções, caso seja implementada uma CDN no ISP (definido pelos autores como μ-CDN), computadas em diferentes janelas de tempo, variando de uma hora a uma semana inteira. Na figura c, é apresentado o tempo necessário para que o cache atinja determinados percentis. Por exemplo, 90% da redução de tráfego é atingida em menos de

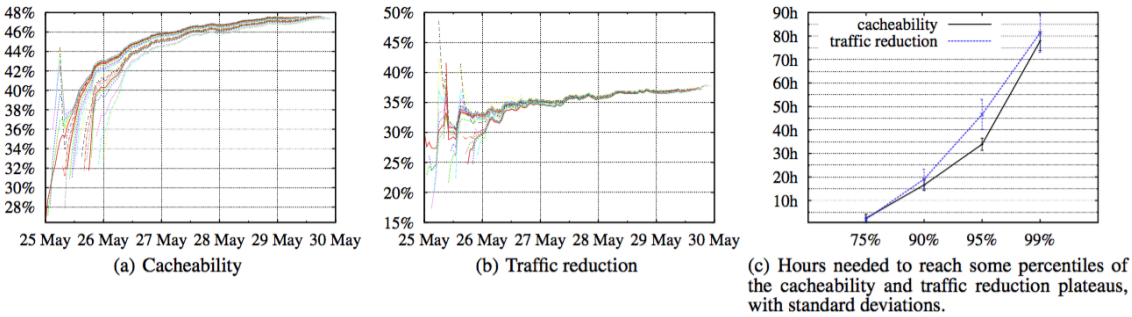


Figura 4.7. Conteúdo cacheável cumulativos e redução do tráfego, a partir de diferentes horas e para várias extensões de tempo. [Imbrenda et al. 2014].

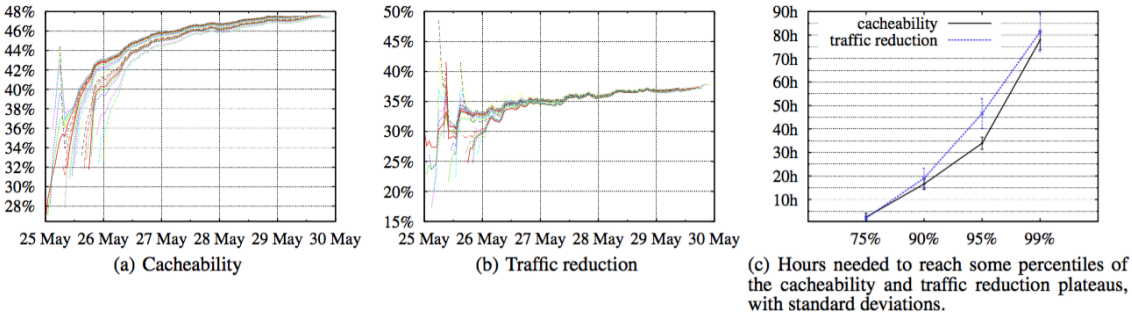


Figura 4.8. Distribuição de requisições por tipo de conteúdo [Shafiq et al. 2014].

24 horas, independente do momento de início. Pelas Figuras 4.7(a) e 4.7(b) observamos que o conteúdo cacheável se estabiliza em torno de 47%, enquanto a redução do tráfego é de praticamente 37%. Esses valores dão uma ideia das oportunidades de cache para um perfil de dados existente atualmente em redes de ISPs.

O sentimento de que há uma grande quantidade de conteúdo *cacheável* também pode ser confirmada a partir do ponto de vista de pontos de presença das CDNs. Em [Shafiq et al. 2014], os autores apresentam um trabalho usando um conjunto de dados de 2012, coletados a partir de múltiplos pontos de presença de um provedor CDN. De fato, de acordo com a Figura 4.8, mais de 40% dos objetos são requisitados mais de uma vez. Ou seja, para esses objetios, o uso de cache CDN pode trazer benefícios recorrentes. A Figura 4.8 também mostra a função de distribuição acumulada por pedidos para diferentes tipos de conteúdo (arquivos). Tais pedidos apresentam características de cauda longa, para os diversos tipos de conteúdo avaliados. A distribuição apresentada enfatiza que uma porção pequena de objetos aparecem na maioria das requisições, enquanto a grande maioria dos objetos são requisitados infrequentemente. Note também uma grande variação de conteúdo. entre os diversos tipos de objetos requisitados.

Notoriamente, YouTube é um dos sistemas Internet atual que posam como grandes

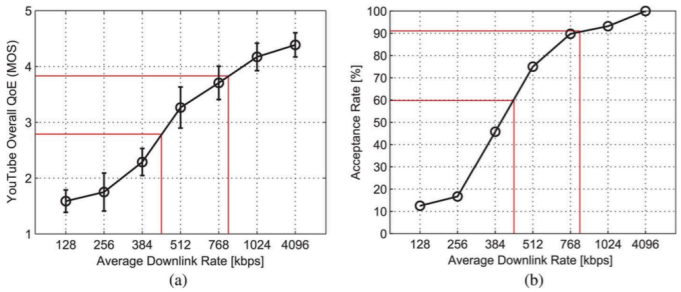


Figura 4.9. QoE e aceitabilidade do YouTube de forma geral com relação a taxa de download [Casas et al. 2014].

vilões, frente aos operadores de rede. Sua enorme popularidade, seu tráfego pesado e a sua complexa infraestrutura distribuída são fatores desafiadores para as operadoras de rede [Casas et al. 2014]. Esses têm que planejar os sistemas de rede apropriadamente, de tal forma que possam se adequar a esse enorme volume de dados e o imprevisível número de clientes do YouTube. Sem dúvida, sistemas como YouTube só conseguem atingir o grande número de clientes com um conteúdo que demanda muitos recursos de rede através da distribuição de seus pontos de serviço. Novamente, o provedor de serviço tenta posicionar seus servidores e redirecionar as requisições para os que estão mais próximos aos clientes. O principal objetivo é reduzir latência e melhorar a qualidade de experiência (QoE). Vários trabalhos tratam do tráfego e das CDNs do YouTube. Por exemplo, [Casas et al. 2014] realizam uma caracterização em larga escala do YouTube em termos de seu tráfego e do provisionamento de seus servidores CDN. Eles consideram problemas de detectar e diagnosticar degradações importantes relacionadas a QoE, dado o tráfego do YouTube, a partir de medidas em ISPs.

Com relação a QoE, os autores conduzem experimentos para verificar a taxa de vídeo recebida por usuários e o número de quebras na visualização do vídeo. A Figura 4.9 é utilizada em [Casas et al. 2014] para mostrar a QoE geral e o aceitável, como declarados pelos usuários assistindo vídeos do YouTube em um teste prático (ambos em função da taxa de download média). Esse experimento foi conduzido ao longo de um mês, onde os usuários assistiam vídeos e suas condições de rede eram alteradas. De acordo com a Figura 4.9(a), mostrando 5 pontos na escala MOS (onde 1 corresponde a QoE muito ruim e 5 como QoE ótimo), há uma queda dessa métrica próximo a 800 kbps (MOS próximo a 4) e a 470 kbps (MOS menor que 3K). O mesmo acontece para a taxa aceitável de serviço (Figura 4.9(b)). Assim, os autores sugerem haver um valor limiar entre ruim e bom entre taxas de 400 e 800 kbps.

Como dissemos anteriormente, o YouTube usa a infraestrutura de CDNs do Google para atender seus cliente. Ele também usa os servidores DNS do Google para redirecionar as requisições entre os servidores e, adicionalmente, eles usam políticas dinâmica de seleção de cache e balanceamento de carga. A Figura 4.10 exemplifica tal estrutura, mostrando o número de IPs únicos servindo o YouTube, assim como os ASes que mantêm o maior número de servidores de compartilhamento. A tabela também mostra o número de IPs por prefixos de rede diferente. Dois ASes do Google são responsáveis pela maior

AS	# IPs	#/24	#/16	% bytes	% flows
All server IPs	3646	97	22	100	100
15169 (Google)	2272	60	2	80.8	77.3
43515 (YouTube)	1222	12	1	19.1	22.5
36040 (YouTube)	43	2	2	< 0.1	< 0.2

Figura 4.10. Número de IPs relacionados a *hosts* do YouTube. O AS 15169 hospeda os caches preferenciais do YouTube [Casas et al. 2014].

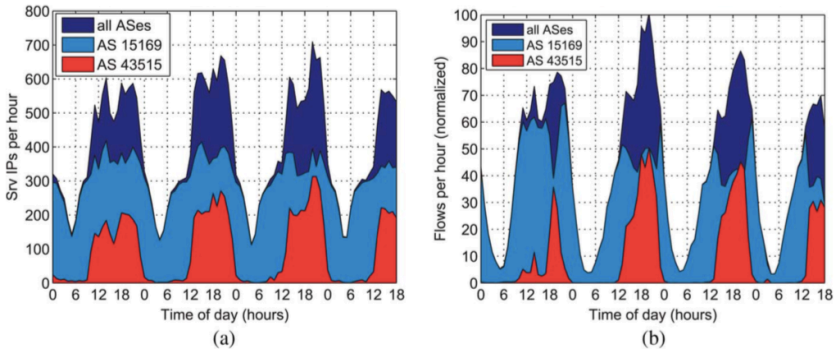


Figura 4.11. IPs e fluxos por hora. Mais de 700 IPs diferentes servem o YouTube em momentos de pico. [Casas et al. 2014].

parte desses IPs (i.e., AS 15169 e AS 43515), agrupados em redes menores /16. Cerca de 80% dos fluxos YouTube são servidos pelo AS 15169, enquanto servidores no AS 43515 são usados para complementar o envio de vídeo aos clientes nas redes monitoradas. A Figura 4.11 mostra a dinâmica do provisionamento de tráfego dos IPs e ASes mencionados. Há mais de 700 IPs ativos servindo o YouTube em momentos de pico. IPs pertencendo ao AS43515 ou AS 15169 apresentam um crescimento em um momento específico do dia. Por exemplo 200 IPs do AS43515 se tornam ativo as 10AM.

Além de métricas que demonstram a distribuição de carga entre servidores da CDN do Google, os autores também mostram dados relativos aos fluxos de rede. A Figura 4.12 mostra a distribuição da vazão média do download de vídeos do youtube. De acordo com essa figura, mais de 30% dos fluxos atingem uma vazão acima de 1 Mbps, enquanto mais de 15% dos fluxos conseguem atingir taxas superiores a 2 Mbps.

Há propostas novas para melhorar os sistemas de distribuição de conteúdo. Por exemplo, enquanto sistemas tradicionais identificam os clientes pelo IP dos seus sistemas DNS, [Chen et al. 2015] propõem um sistema que usa diretamente o prefixo do IP do cliente para identificá-los. O principal problema que esses novo sistema resolve é: sistemas tradicionais dependem da infraestrutura de servidores DNS para localizar qual de seus servidores irá tratar uma requisição. A premissa que usuários usam seus servidores locais (i.e. no mesmo local físico ou rede) não é mais verdadeira. De fato, é bem comum que usuários Internet configurem seus sistemas para utilizar servidores DNS externo, como

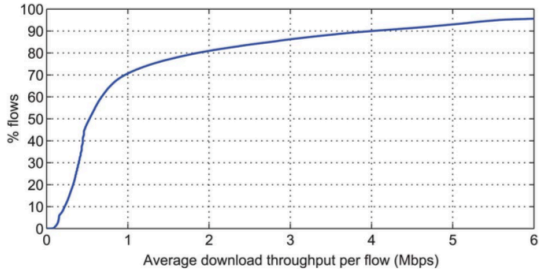


Figura 4.12. Vazão do fluxo de *downlink* médio do YouTube [Casas et al. 2014].

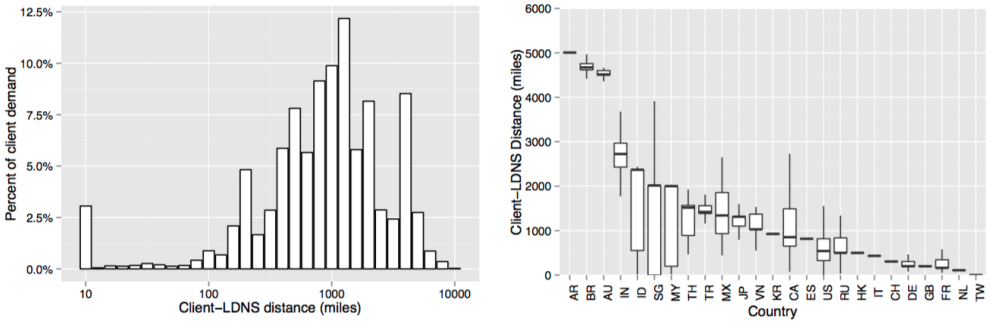


Figura 4.13. Histograma da distância para clientes que usam servidores públicos [Chen et al. 2015]: (esquerda) visão global; (direita) visão detalhada por país.

o servidor DNS público do Google. Ao quebrar a premissa sob qual sistemas CDNs foram construídos, um cliente pode ser direcionado a um servidor afastado dele. Imagine a situação que, um cliente usa um servidor DNS público do Google. Mesmo estando localizado no Brasil, o servidor DNS vai retornar ao cliente um IP de servidor que irá tratar sua requisição mais próximo ao DNS. Se a Akamai exportou um IP para o DNS do Google de um servidor nos EUA, o cliente brasileiro será tratado por um servidor inapropriado à sua localização geográfica.

O problema descrito acima fica bem claro pela Figura 4.13 (esquerda), onde é apresentado um histograma da distância para clientes que usam servidores públicos. Notamos que muitos clientes estão com distâncias notoriamente maiores do que a população como um todo. Isso reflete o fato que servidores públicos DNS nem sempre estão no mesmo local que o cliente. Ao se realizar uma quebra por país, de acordo com a Figura 4.13 (direita), verificamos distâncias ainda mais desproporcionais para clientes usando DNS públicos em algumas regiões, como América do Sul, Ásia e Oceania. Argentina e Brasil são os exemplos com maior disparidade. Nesses países, o servidor público DNS é do Google e, presume-se que não há presença local de tais servidores. Note que Singapura e Malásia são bem servidos por servidores públicos DNS hospedados em Singapura. Porém, devido a acordos de *peering*, muitos clientes nesses países são roteados para servidores DNS mais distantes.

Tabela 5.1. Perfil de ASes nos PTTs do Brasil [Brito et al. 2015].

Classificação	Brasil (*)	DF	MG	RS	SP	VIX
1. Provedor de Internet	65,1% ± 20%	37,5%	55,9%	68,0%	73,1%	75,0%
1.1 Provedor de Trânsito	8,6% ± 09%	20,8%	14,7%	5,0%	5,6%	10,0%
1.2 Provedor de Acesso	56,5% ± 21%	16,7%	41,2%	63,0%	67,5%	65,0%
2. Provedor de Serviços	10,1% ± 07%	8,3%	8,8%	5,0%	12,5%	5,0%
2.1 Provedor de Conteúdo	3,2% ± 06%	0,0%	2,9%	3,0%	4,7%	0,0%
2.2 Provedor de Hospedagem	6,8% ± 05%	8,3%	5,9%	2,0%	7,8%	5,0%
3. Organização Pública	12,3% ± 21%	37,5%	20,6%	11,0%	4,4%	15,0%
3.1 Universidade Pública	1,8% ± 19%	0,0%	0,0%	2,0%	1,1%	0,0%
3.2 Governo	8,8% ± 13%	33,3%	17,6%	8,0%	2,2%	15,0%
3.3 Outros	1,8% ± 03%	4,2%	2,9%	1,0%	1,1%	0,0%
4. Organização Privada	12,6% ± 09%	16,7%	14,7%	16,0%	10,0%	5,0%
4.1 Universidade Privada	0,7% ± 03%	0,0%	2,9%	4,0%	0,0%	0,0%
4.2 Empresa Privada	10,4% ± 09%	16,7%	8,8%	10,0%	8,9%	5,0%
4.3 Outros	1,5% ± 09%	0,0%	2,9%	2,0%	1,1%	0,0%

(*) Média de todos os 26 PTTs brasileiros.

4.5.3. Tráfego e roteamento

Apesar da sociedade considerar a Internet como uma infraestrutura critica agora, nós ainda temos pouco conhecimento sobre a dinamicidade de sua estrutura [Lodhi et al. 2014]. O entendimento da Internet como um todo é desafiador. Há um número inimaginável de elementos conectados, com mudanças constantes em seus relacionamentos. Porém, pontos de troca de tráfego (PTTs) na Internet podem retratar a rede como um todo de forma interessante. Primeiro, eles representam um microcosmo da diversidade da Internet [Brito et al. 2015]. Segundo, o desenvolvimento atual da Internet está fortemente relacionado a tais estruturas.

Mais precisamente, um ponto de troca é uma forma de promover conectividade entre diversos sistemas autônomos. Nesses PTTs, dezenas ou mesmo centenas de ASes realizam parcerias (*peering*) e assim, conseguem diminuir a distância fim-a-fim na Internet. Como principal consequência notada, usuários Internet conseguem acessar serviços de forma mais rápida. Porém, os principais benefícios giram em torno de desempenho (da rede) e de segurança [Norton 2011]. Por exemplo, durante períodos de congestionamento, decorrentes de tráfego de ataques de negação de serviço, um outro tráfego direto através de *peering* em PTTs está separado e não sofre das vulnerabilidades decorrentes do primeiro.

Em suma, o entendimento da dinâmica em um PTT pode auxiliar tanto pesquisadores quanto administradores de rede a criarem melhores protocolos e serviços. Nesse sentido, [Brito et al. 2015] apresentam um estudo sobre PTTs brasileiros. Eles realizaram uma compilação ampla de dados, a partir de conexões telnet com todos os Looking Glasses (LG) dos PTTs brasileiro. Os autores coletam dados como a tabela de rotas BGP do plano de controle e relação de AS-PATH do BGP. Em alguns casos, os autores solicitaram dados diretamente ao NIC.br.[16]

Um resumo do resultados de [Brito et al. 2015] pode ser visto na Tabela 5.1. A maior parcela parcela dos membros interessados em *peering* consiste em provedores de acesso de abrangência regional. Tais provedores têm interesses econômico por trás: eles

[16]http://www.nic.br

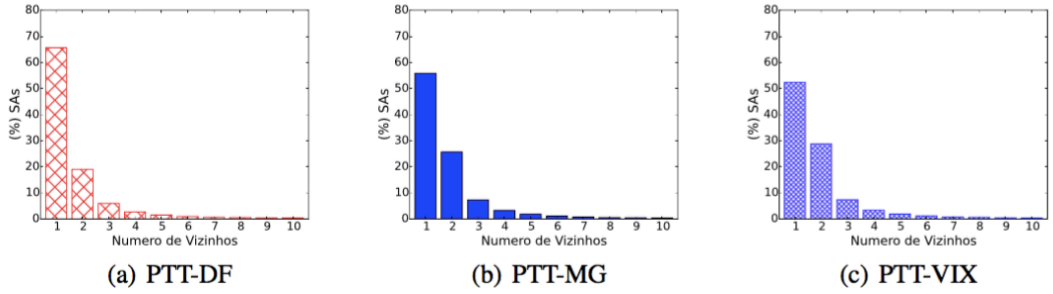


Figura 4.14. Distribuição dos graus [Brito et al. 2015].

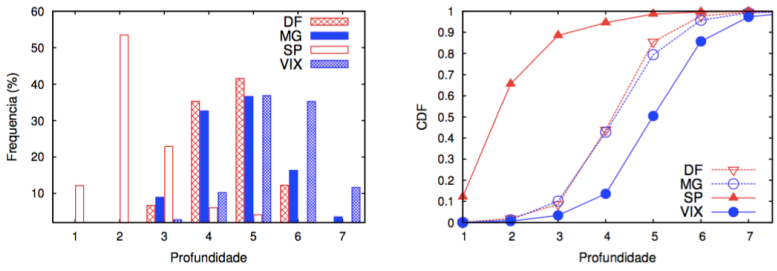
esperam trocar o máximo de tráfego possível através de acordos multilaterais na infraestrutura compartilhada dos PTTs, uma vez que dessa maneira há economia decorrente da não utilização do trânsito provido pelas grandes operadoras de telecomunicações. Ainda de acordo com a Tabela 5.1, é possível perceber que há uma baixa participação dos provedores de conteúdo nas diversas regiões brasileiras (e.g. jornais, revistas, rádios, emissoras de televisão, etc). A maioria dos representantes da classe provedor de conteúdo são empresas que operam Content Delivery/Distribution Networks (CDN). Da mesma forma, há baixa presença de empresas privadas nos PTTs.

A maioria absoluta dos SAs autônomos (98%) conectados aos PTTs optaram por *peering* aberto com acordo de tráfego multilateral (ATM). Praticamente todos os demais ASes que optaram por *peering* privado através de acordos de tráfego bilateral (ATB) são provedores de trânsito, a destacar (e.g. Brasil Telecom, Embratel, Global Crossing, Level3, NTT, Oi, Telefonica e TIM). Em [Brito et al. 2015], os autores remontam o grafo de conexão dos ASes que se conectam aos PTTs. Assim, eles conseguem inferir importantes métricas acerca da topologia de tal microsistema Internet. Por exemplo, de acordo com a Figura 4.14, a grande maioria dos ASes tem grau baixo. Usualmente, ASes com maior grau são operadoras de telecomunicações, que têm mais adjacências pela natureza da atividade de venda de trânsito para provedores de acesso.

Assim como o grau, a profundidade ou diâmetro dos anúncios observados na tabela BGP dos PTTs, com base no atributo AS-PATH, são pequenos (Figura 4.15). Note que, uma profundidade igual a 1 indica que o AS-PATH é composto de apenas um AS, o que deve ser interpretado como diretamente conectados no PTT. Tomando o PTT-VIX na Figura 4.15(a) como referência, por exemplo, lê-se que aproximadamente 40% (eixo y) do total de rotas anunciadas na tabela BGP possuem 5 ASes e outros 40% possuem 6 ASes no atributo AS-PATH, ou seja, cerca de 80% das rotas têm profundidade de apenas 5 ou 6 ASes.

Além de consultas diretas a Looking Glasses, há outras maneiras de inferir a dinamicidade e a topologia existente em PTTs. Por exemplo, [Lodhi et al. 2014] utilizam o PeeringDB[17] como fonte de seu estudo. O PeeringDB é um banco de dados online onde operadores de ASes provêem informações acerca das redes, como por exemplo, políticas de *peering*, volume de tráfego e localizações geográficas.

[17]https://www.peeringdb.com/



(a) Profundidade Isolada (b) Profundidade Acumulada

Figura 4.15. Profundidade dos caminhos anunciados (AS-Path) [Brito et al. 2015].

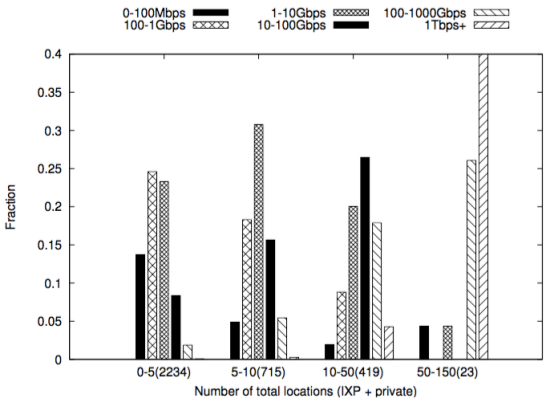


Figura 4.16. Número de PTTs e locais de peering privados onde as redes avaliadas estão presentes *versus* o volume de tráfego reportado [Lodhi et al. 2014].

De acordo com [Lodhi et al. 2014], provedores de serviço de redes estão presentes em mais PTTs e instalações de *peering* privados, comparado aos demais tipos de redes. Diferente do que ocorre no trabalho que analisa os PTTs brasileiros [Brito et al. 2015], a presença de companhias em instalações de *peering* privados é alta, comparável ao número de provedores de conteúdo e acesso. Segundo os autores, a amostragem de redes de companhias no PeeringDB é pequena (apenas 120 redes) e contém redes como Amazon e Websense, que são *peers* em muitas outras localizações. Isso sugere que há uma tendência de *peerings* mais ricos na periferia da Internet.

A Figura 4.16 mostra o número de localizações que uma rede está presente e mostra a distribuição do volume de tráfego. No geral, o número de localizações em que uma rede está presente correlaciona-se com o volume de tráfego anunciado. A fração de redes anunciando grandes volumes de tráfego (100-1000Gbps e 1Tbps+) cresce com o total de localizações. O número de localizações de *peerings* de uma rede é usualmente mais fácil de se conhecer do que o tráfego dela e a correlação desses dois fatores sugere uma maneira de estimar, mesmo que grosseiramente, a última métrica, com base na primeira.

A Figura 4.17 apresenta a mediana, bem como os percentis 10 e 90, para os espaços de endereço anunciados para cada volume anunciado. Redes de acesso e coor-

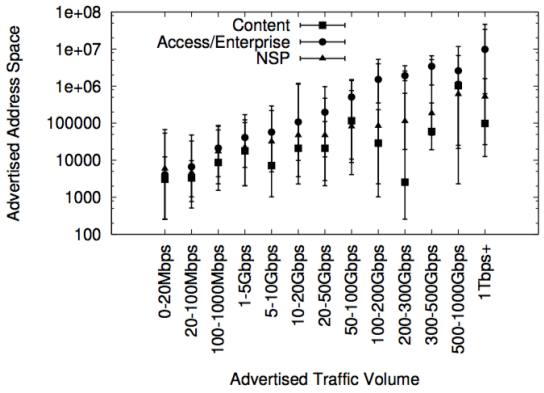


Figura 4.17. Volume de tráfego reportado versus espaço de endereçamento anunciado [Lodhi et al. 2014].

porativas têm uma mediana maior de espaço de endereços anunciados para cada volume de tráfego. Há uma correlação forte nesse caso. Provedores de conteúdo tem medianas menores, com correlação fraca. Os provedores de serviço de redes tem as maiores correlações observadas. mesmo com mediana menores que redes de acesso e corporativas.

Finalmente, destacamos a Figura 4.18, que apresenta a estratégia de *peering* pelo volume de tráfego. Essa figura mostra as estratégias de *peerings* para redes de provedores de serviço, conteúdo e provedores de acesso para um determinado volume de tráfego. Para redes de provedores de serviço, e redes de acesso, o tipo preferencial é por *peering* aberto, que gradualmente deixa de ser o preferencial quando o volume de tráfego do AS cresce. Redes de provedores de serviços com baixo volume e provedores de acesso mostram uma preferência forte por *open peering*; 80% das 415 redes de provedores de serviço e 87% dos 603 provedores de acesso, que anunciam tráfego menores que 5 Gbps, anunciam a política de *peering* aberto.

Por outro lado, apenas 1 de 5 redes de provedores de serviço e 2 em 18 de provedores de acesso que anunciaram mais de 1 Tbps de tráfego declaram *peering* aberto. Provedores de conteúdo apresentam uma relação mais fraca entre tráfego e o tipo de política. Enquanto 88% dos 573 provedores de conteúdo que declaram baixo tráfego aponta *peering* aberto e; 4 dos 8 que anunciaram mais de 1 Tbps de tráfego anunciam *peering* aberto e nenhum deles peering restrito.

4.6. Validações e Testbeds

As iniciativas de medições de desempenho têm sido fortemente adotadas, bem como validadas, em ambientes de experimentação (*testbeds*) da Internet em anos recentes. Em tais ambientes, arcabouços de monitoramento e controle (CMFs) são adotados para coletar e fornecer informações de status e desempenho dos experimentos conduzidos pelos usuários (experimentadores) e das infraestruturas utilizadas pelos mesmos [Marcondes et al. 2012].

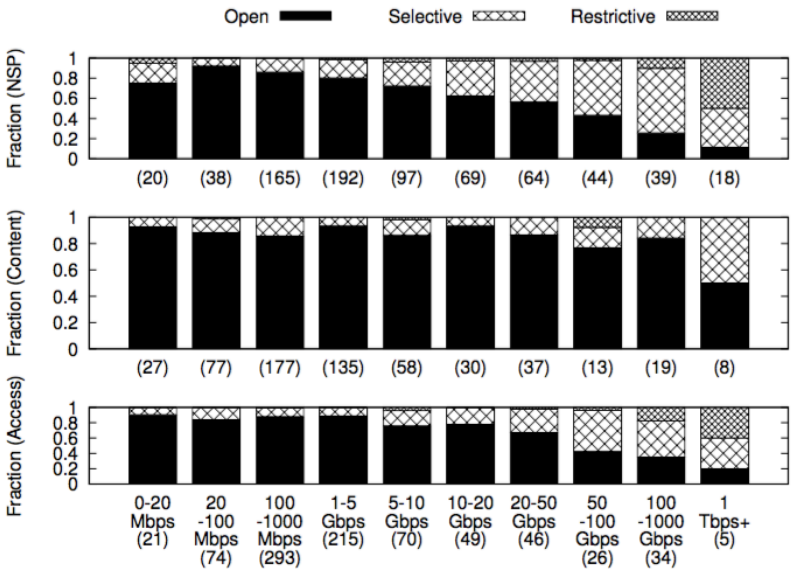


Figura 4.18. Estratégia de *peering* classificado por volume de tráfego [Lodhi et al. 2014].

A partir dos CMFs, as medições são realizadas através de arquiteturas de software que oferecem recursos para realização de testes de desempenho, armazenamento de dados, orquestração de serviços, acesso, visualização e análise de dados [Marcondes et al. 2012, Pinheiro et al. 2012]. Na busca por maior flexibilidade no uso dos componentes responsáveis por cada uma dessas funcionalidades, algumas dessas arquiteturas são projetadas através de um conjunto de serviços medições fracamente acoplados. O baixo acoplamento é obtido a partir do desenvolvimento de interfaces de comunicação padronizadas, implementadas em linguagens como o XML e JSON.

A busca por arquiteturas de software com baixo acoplamento fez com que muitos dos ambientes de experimentação adotassem os serviços disponíveis no arcabouço do PerfSONAR [Hanemann et al. 2005]. Trata-se de um conjunto de serviços de medições que disponibiliza as suas funcionalidades a partir de interfaces auto-descritíveis. Através de metadados, as interfaces fornecem às aplicações usuárias dos serviços os subsídios necessários para processar a informação proveniente do monitoramento. Inicialmente, os serviços disponibilizados no PerfSONAR utilizaram a linguagem XML para implantação dessas interfaces. Atualmente existem iniciativas de desenvolvimento na linguagem JSON. Além do PerfSONAR, outros arcabouços tem sido propostos para auxiliar experimentadores através das medições. Entre eles, podemos citar a biblioteca OML (*Orbit Measurement Library*) [Singh et al. 2005], que tem como principal objetivo oferecer suporte similar ao apresentado pelo PerfSONAR.

Esta seção discute como algumas dessas infraestruturas de experimentação fazem uso das medições. O destaque vai para os arcabouços de controle e monitoramento (CMFs) utilizados pelas arquiteturas PlanetLab [Bourgeau et al. 2011], GENI,[18] OR-

[18] `http://www.geni.net`

BIT[19] [Raychaudhuri et al. 2005], OFELIA [Suñé et al. 2014], FIRE [Al-Hazmi and Magedanz 2012] e FIBRE [Pinheiro et al. 2012]. Uma discussão mais detalhada sobre os principais requisitos de tais arcabouços pode ser encontrada no minicurso "*Estado da arte de sistemas de controle e monitoramento de infraestruturas para experimentação de redes de comunicação*" apresentado na 30ª edição do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) [Marcondes et al. 2012].

4.6.1. PlanetLab

O PlanetLab [Bavier et al. 2004], que teve início em 2003 e se tornou um dos mais antigos e utilizados ambientes de experimentação de redes no mundo, está localizado em diferentes países por meio de um acordo envolvendo diferentes instituições acadêmicas e governamentais de diversos países. O PlanetLab faz um forte uso da abordagem da virtualização como forma de compartilhar os recursos computacionais entre os pesquisadores. Atualmente o PlanetLab consiste em um grande laboratório compartilhado formado por 1353 nós instalados em 717 localidades[20].

Um dos arcabouços utilizados no monitoramento do PlanetLab é o TopHat [Bourgeau et al. 2011]. O TopHat adota a abordagem de medições ativas na obtenção de informações de topologia do *testbed*, permitindo que o usuário (experimentador) possa alocar de forma mais eficiente as fatias e recursos necessárias para a realização dos seus experimentos. Em outras palavras, todo o ciclo de vida do experimento é atendido pela ferramenta da seguinte forma: (i) na etapa de configuração, o TopHat auxilia o usuário escolher os nós de acordo com métricas de desempenho, tais como atraso e número de saltos de uma rota relacionada ao experimento; (ii) em seguida, durante a execução do experimento, a ferramenta permite ao usuário monitorar o desempenho, fazer medições e modificar o experimento a partir das informações obtidas; e (iii) por fim, após a conclusão do experimento, o usuário pode visualizar dados da monitoração armazenados em base de dados.

4.6.2. GENI

O GENI (*Global Environment for Network Innovations*) consiste em um ambiente de experimentação americano financiado pela NSF (*National Science Foundation*).[21] Os princípios fundamentais do GENI são a programabilidade, a experimentação baseada em fatias, a virtualização e, por fim, a federação. A programabilidade tem o propósito de oferecer aos pesquisadores a possibilidade de instalar seus próprios códigos de software e controlar seus comportamentos. A experimentação baseada em fatias (*slice-based*) visa permitir a utilização conjunta de recursos computacionais (ex. CPU, memória, disco, etc.) de diferentes localidades. A virtualização tem como objetivo fazer com que os mesmos recursos computacionais sejam utilizados de forma compartilhada e por mais de um experimento. Por fim, a federação procura fazer com que o experimentador tenha acesso aos recursos pertencentes a diferentes domínios administrativos de rede.

Por conta de tais princípios, no que diz respeito aos arcabouços de monitoração e

[19] `http://www.orbit-lab.org`

[20] Dados obtidos no período em que este documento foi elaborado

[21] `http://www.nsf.gov/`

controle (CMFs), o GENI especifica um conjunto bem amplo de requisitos e uma arquitetura orientada a serviços que serve de guia para implantação dos sistemas de monitoração utilizados pelos usuários. A arquitetura proposta se baseia nos serviços já propostos pelo PerfSONAR, oferecendo serviços de: (i) orquestração de medições (MO – *Measurement Orchestration*); (ii) ponto de medição (MP – *Measurement Point*); (iii) informação de medições (MI – *Measurement Information*); (iv) coleta de dados (MC – *Measurement Collection*); (v) apresentação e análise (MAP – *Measurement Analysis and Presentation*); e (vi) armazenamento de dados (MDA – *Measurement Data Archive*). Esse grupo de serviços, em conjunto com os requisitos especificados para a arquitetura, serviu de base para a criação de duas infraestruturas de medições derivadas: O GEMINI (*GENI Measurement and Instrumentation Infrastructure*) e o GIMI I&M System. O GEMINI é um desenvolvimento resultante da parceria entre as Universidades de Indiana e Kentucky. O arcabouço faz uso das ferramentas do pacote INSTOOLS para fazer medições passivas através do SNMP e no pacote LAMP com o objetivo de promover medições ativas no ambiente através das ferramentas OWAMP, BWCTL, Ganglia e PingER. Já o sistema GIMI é resultante de um projeto desenvolvido pela universidade de Massachusetts e os institutos de pesquisa NICTA[22] e RENCI.[23]

4.6.3. OMF e outros ambientes para experimentações sem fio

Entre os ambientes de experimentação em redes sem fio, destaca-se o ORBIT (*Open Access Research Testbed for Next-Generation Wireless Networks*)[24] [Raychaudhuri et al. 2005]. Esse *testbed* apresenta um ambiente experimental *indoor*, multiusuário e flexível para dar suporte a pesquisas de redes sem fio de nova geração. Para o ambiente do ORBIT foi desenvolvido o arcabouço OMF (*ORBIT Management Framework*) [Rakotoarivelo et al. 2010] pela Universidade Rutgers e o instituto NICTA (*National ICT Australia*). O OMF consiste num arcabouço de controle e monitoramento de ambientes de experimentação originalmente planejado para ser utilizado com redes sem fio do projeto ORBIT, mas que atualmente tem sido utilizado por outros *testbeds*.

O OMF beneficia essencialmente os experimentadores na medida em que fornece um conjunto de ferramentas para especificar, implementar e executar experimentos. Tais ações são realizadas a partir de funcionalidades projetadas para: (i) descrição, gerenciamento e controle de experimentos; (ii) gerenciamento e controle de recursos e agregados; e (iii) coleta de dados de medições. Além disso, o OMF disponibiliza a biblioteca OML (*OMF Measurement Library*) [Singh et al. 2005]. Essa biblioteca facilita o desenvolvimento de aplicações de monitoramento para ambientes de rede sem fio. A biblioteca permite a geração de códigos em C/C++ que podem ser compilados na aplicação cliente (nós participantes do experimento) para definir parâmetros e pontos de medição, coleta de dados e armazenamento em banco de dados.

Além da plataforma Orbit, que se tornou bastante conhecida e popular para medições e experimentação com foco específico em redes sem fio, há outros ambientes com finalidades semelhantes. Um exemplo mais recente é a plataforma AirLab [Kone et al. 2011]. AirLab provê uma infraestrutura para coleta de dados de redes sem fio de forma

[22] `http://nicta.com.au`
[23] `http://www.renci.org`
[24] `http://www.orbit-lab.org`

distribuída baseada em múltiplos nós distribuídos geograficamente para coletar *traces* em experimentos padronizados ou definidos pelo usuário. Uma outra iniciativa relacionada foi o SensLAB [des Roziers et al. 2011] que permite a avaliação de protocolos e aplicações em redes de sensores sem fio de forma escalável. A evolução do SensLAB culminou na plataforma experimental de Internet das Coisas IoT-LAB [Fleury et al. 2015],[25] que hoje integra a iniciativa do OneLab.[26] OneLab é uma federação de *testbeds* que inclui plataformas de experimentação em diferentes áreas de redes, tais como Internet das Coisas, Internet, Redes Definidas por Software, computação em nuvem e também redes sem fio. Um *survey* recente por [Goel et al. 2016] fornece uma visão geral de novos *testbeds*, ferramentas e serviços focados em medição do desempenho fim-a-fim de redes móveis.

4.6.4. OFELIA

O OFELIA (*OpenFlow in Europe: Linking Infrastructure and Applications*) [Suñé et al. 2014] consistiu em um projeto europeu, descontinuado em 2013, que apresentava à comunidade de pesquisa recursos para experimentação e controle de experimentos de redes. O projeto se baseou no uso do Openflow para prover a virtualização e controle dos recursos de rede através de interfaces padronizadas e de fácil acesso. Em termos de infraestrutura de medições, o projeto desenvolveu o arcabouço OCF (*Ofelia Control Framework*).

O arcabouço OCF faz uso dos sistemas OFLOPS [Rotsos et al. 2012] e Cbench [Tootoonchian et al. 2012] para monitoramento da protocolo Openflow. O primeiro é utilizado para verificar o desempenho de comutadores Openflow. A abordagem da ferramenta consiste em emular o comportamento do controlador interagindo com os comutadores a fim de obter informações de desempenho de métricas do SNMP e do próprio Openflow. Além disso, a ferramenta pode ser utilizada para a geração de pacotes utilizados também para a medição de desempenho dos equipamentos. Já o Cbench é utilizado para avaliar o desempenho do controlador a partir da emulação de requisições aos comutadores e envio das respostas a um controlador alvo que tem seu desempenho avaliado. As métricas avaliadas no controlador são a latência para configuração de fluxos e o número de fluxos instalados por segundo.

4.6.5. FIRE

A iniciativa FIRE (*Future Internet Research and Experimentation*) [Schaffers 2015] está entre os projetos para construção de ambientes de experimentação mais significativos na Europa. A iniciativa FIRE engloba diversos projetos orbitais, com objetivos mais específicos no que diz respeito à prática de experimentação em redes. Entre os projetos orbitais do FIRE, destaca-se o Fed4FIRE (*Federation for FIRE - Future Internet Research and Experimentation*). Iniciado em 2012, com finalização prevista para setembro de 2016, o Fed4FIRE tem como objetivo principal criar um ambiente de federação dos recursos de experimentação de diversos outros ambientes de experimentação existentes.[27]

Um dos arcabouços de monitoração e controle utilizados no FIRE é o MOST4FIRE

[25] `http://www.iot-lab.info`

[26] `http://onelab.eu`

[27] Uma lista completa dos projetos relacionados ao FIRE encontra-se em: `http://doc.fed4fire.eu/testbeds.html`

[Al-Hazmi and Magedanz 2012]. O arbouço tem como objetivo o gerenciamento dos recursos de monitoração de ambientes de experimentação federados, caracterizados por infraestruturas heterogêneas. O conceito principal da ferramenta é oferecer o monitoramento como um serviço a partir de uma representação unificada dos dados obtidos de variadas fontes ou através de um conjunto de ferramentas de monitoração, que com base no conjunto de requisitos de monitoramento descritos, são apresentadas aos usuários. Tais ferramentas podem ser acessadas diretamente pelas suas interfaces ou através de interfaces de programação (API).

4.6.6. Redes experimentais brasileiras

No Brasil, ao longo dos últimos 15 anos, a Rede Nacional de Ensino e Pesquisa (RNP) tem liderado diversas iniciativas de medições através de grupos de trabalho e projetos de prospecção e desenvolvimento tecnológico [Sampaio et al. 2007]. Muitas dessas iniciativas estiveram inseridas no contexto de ambientes de experimentação, através dos quais foram desenvolvidos arcabouços de monitoramento e utilizadas diversas ferramentas de medição. A iniciativa de maior destaque foi a colaboração da RNP no desenvolvimento do arcabouço de monitoramento PerfSONAR. Assim como aconteceu em projetos de outras instituições internacionais, o PerfSONAR foi utilizado em muitos projetos de pesquisa no Brasil apoiados pela RNP. A RNP também faz uso do PerfSONAR para auxiliar as atividades de operação da rede IPê, por meio do serviço monitoramento MonIPê. O Comitê Técnico de Monitoramento de Redes (CT-Mon) da RNP é o fórum responsável pela prospecção tecnológica na área de medições de rede, visando por exemplo a estruturação de recomendações estratégicas para a evolução do serviço MonIPê da RNP.

Mais recentemente, diante da percepção sobre a importância das redes experimentais programáveis e sua forte expansão em todo o mundo, a RNP, em conjunto com diversas instituições de ensino e pesquisa do Brasil, envidou esforços no sentido de desenvolver uma rede experimental brasileira, através do projeto FIBRE (Experimentação no Futuro da Internet entre BRasil e Europa). O projeto, aprovado na primeira Chamada Coordenada BR-EU (Edital MCT/CNPq Nº.066/2010), teve o propósito de implantar e validar uma infraestrutura de experimentação compartilhada de larga escala de forma a ampliar a colaboração entre pesquisadores brasileiros e europeus. No lado brasileiro, pesquisadores possuem acesso à rede do FIBRE (FIBRE-NET) através das instituições vinculadas à rede Ipê da RNP. Nessa rede, recursos computacionais (nós sem fio, comutadores e máquinas virtuais) são disponibilizados a todos os usuários através de ilhas experimentais localizadas nos laboratórios de cada instituição parceira do projeto. Essas ilhas oferecem um ambiente de experimentação local que permite aos pesquisadores e professores de instituições de ensino e pesquisa a promover inovações e capacitar recursos humanos no melhor entendimento das redes de computadores e suas aplicações.

No projeto FIBRE, muitos dos arcabouços discutidos nas seções anteriores foram avaliados e implantados de forma conjunta no ambiente de experimentação, sobretudo porque cada ilha de experimentação possui um conjunto diversificado de recursos computacionais, tais como nós sem fio. Portanto, os principais arcabouços utilizados no FIBRE foram: OMF, OCF, e ProtoGENI. Para isso, foi necessário realizar customizações em tais arcabouços de modo que esses pudessem ser utilizados numa mesma arquitetura de software (FIBRE-BR I&M) de forma transparente ao usuário final [Pinheiro et al. 2012]. Por

conta da heterogeneidade de soluções e da experiência obtida em projetos anteriores, foi utilizado os serviços do PerfSONAR nesta customização, onde foram desenvolvidos os serviços de orquestração, integração de dados de medição (*MDIP – Measurement Data Integration Points*), portal de visualização, e armazenamento persistente de dados.

Destaca-se também no Brasil a arquitetura FITS (*Future Internet Testbed with Security*)[28] [Moraes et al. 2014]. Trata-se de uma arquitetura flexível para experimentação de redes de propósito geral que apresenta aos experimentadores um ambiente de avaliação de protocolos de redes e novas propostas para a Internet do Futuro que se baseiem no uso do Openflow e a plataforma de virtualização Xen. A rede física do *testbed* FITS abrange instituições brasileiras e européias. Em termos de medições, a rede FITS é equipada com múltiplos pontos de medições utilizados no monitoramento dos nós físicos, disponibilizados aos usuários para os experimentos. As informações provenientes desse monitoramento auxiliam os usuários na alocação dos recursos. Nesse sentido, através de uma plataforma web, a FITS fornece aos usuários informações de throughput e RTT (*round-trip time*) de canais virtuais e físicos, dados de topologia, dados relacionados aos roteadores virtuais e informações de fluxos.

Dois novos projetos de redes experimentais foram aprovados recentemente e estão em fase de implantação. O primeiro consiste no projeto FUTEBOL (*Federated Union of Telecommunications Research Facilities for an EU-Brasil Open Laboratory*)[29], aprovado em resposta à terceira chamada pública coordenada BR-UE em Tecnologias da Informação e Comunicação (TIC). O projeto tem o objetivo de criar uma rede de teste para experimentação de soluções de integração de redes ópticas e sem fio. O FUTEBOL possui colaborações com os projetos FIBRE, no Brasil, Fed4FIRE e FIRE na Europa. O segundo projeto, de menor escopo, se trata do desenvolvimento da rede BAMBU [Sampaio et al. 2015]. Uma rede experimental metropolitana para a cidade de Salvador que, em parceria com outros projetos em andamento (ex. FIBRE e FUTEBOL), expandirá as possibilidades de experimentações dos pesquisadores locais. Em ambos os projetos, iniciativas de medições estão previstas.

4.7. Considerações Finais

A Internet expandiu-se enormemente tanto em abrangência quanto em diversidade. Isso gerou grande demanda para expansão também da área de Metrologia de Redes para acompanhar essas novas tendências, permitindo a caracterização, análise e modelagem dos diferentes contextos em que redes se fazem presentes hoje em dia. Ao longo deste documento revisitamos a área de Metrologia de Redes para focar em seus avanços na última década, cobrindo sua utilização (i) em diferentes infraestruturas (p.ex. redes sem fio e redes virtualizadas), (ii) sob a perspectiva dos usuários finais, (iii) para caracterização e análise do tráfego de rede; e (iv) em *testbeds* recentes para validação de propostas.

Enquanto a Internet se expande e se diversifica, os desafios para a realização de medições e seu monitoramento também se multiplicam. O volume de dados a tratar, inclusive de fontes heterogêneas, também é crescente, o que impõe novos desafios para sua coleta, tratamento, processamento e análise. Assim, há uma clara tendência na busca

[28] `http://www.gta.ufrj.br/fits`
[29] http://www.ict-futebol.org.br/

de novos métodos acerca de metrologia, para fazer frente a esses desafios. Tais métodos devem ser eficientes tanto em capacidade de lidar com tais volumes de dados bem como para processá-los.

Uma área de crescente atividade é a aplicação de técnicas mineração de dados e de aprendizagem de máquina (*machine learning*) em dados de rede. Há um grande interesse, por exemplo, no uso de tais técnicas em análise preditiva ou detecção de anomalias. Nesse contexto, métodos de aprendizagem de máquina podem ser aplicados para melhorar o controle e gerenciamento de aplicações de rede e seus serviços. Isso inclui mecanismos de aprendizagem de máquina para a aquisição de conhecimento a partir de redes existentes de forma que novas redes possam ser estabelecidas com menor esforço, o controle e otimização de roteamento, para gerenciar a rede de forma mais autonômica e predizer estados futuros da rede. Ações de pesquisa e desenvolvimento nessa linha estão no centro de iniciativas recentes, tais como o recém-proposto grupo de pesquisa em Aprendizagem de Máquina para Redes (*N*etwork Machine Learning Research Group – NMLRG)[30] do IRTF.

Dadas as necessidades de gerenciamento e segurança de redes em relação a reações rápidas e eficazes, cresce a demanda por métodos de processamento de grandes volumes de dados de redes de forma eficiente. Nesse contexto, há uma tendência também de investigação de métodos para processamento em memória (*in-memory*) de fluxos de dados de rede. Há desafios para a amostragem e correlação de dados de fontes heterogêneas, por exemplo, para detecção de anomalias ou recomendações de procedimentos para a operação da rede.

Enfim, a área de Metrologia de Redes é bastante abrangente. Mesmo cobrindo diversos aspectos relevantes, ainda alguns tópicos importantes não foram contemplados em detalhes nesse trabalho. Por exemplo, não foi contemplada aplicação de Metrologia de Redes para monitoramento de violações de segurança, seja por ataques, ações maliciosas ou quebra de privacidade. Possivelmente, o foco devido na área de Metrologia de Redes aplicada à cibersegurança seria merecedora de um minicurso próprio dada sua abrangência atual. Preferimos focar o minicurso em aspectos relacionados à caracterização e desempenho da rede em diferentes contextos atuais.

Agradecimentos

Os autores são gratos ao apoio do CNPq, FINEP, CT-Mon/RNP, CGI.br e FAPERJ.

Referências

[Adami et al. 2012] Adami, D., Callegari, C., Giordano, S., Pagano, M., and Pepe, T. (2012). Skype-hunter: A real-time system for the detection and classification of skype traffic. *Int. J. Communication Systems*, 25(3):386–403.

[Adamic and Huberman 2000] Adamic, L. A. and Huberman, B. A. (2000). Power-law distribution of the World Wide Web. *Science*, 287(5461):2115–2115.

[Adhikari et al. 2015] Adhikari, V., Guo, Y., Hao, F., Hilt, V., Zhang, Z.-L., Varvello, M., and Steiner, M. (2015). Measurement study of netflix, hulu, and a tale of three cdns. *Networking, IEEE/ACM Transactions on*, 23(6):1984–1997.

[30] `http://datatracker.ietf.org/doc/charter-irtf-nmlrg/`

[Adhikari et al. 2012a] Adhikari, V., Jain, S., Chen, Y., and Zhang, Z.-L. (2012a). Vivisecting youtube: An active measurement study. In *INFOCOM, 2012 Proceedings IEEE*, pages 2521–2525.

[Adhikari et al. 2012b] Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., and Zhang, Z. L. (2012b). Unreeling netflix: Understanding and improving multi-CDN movie delivery. In *Proc. of the IEEE INFOCOM*, pages 1620–1628, Orlando, FL, USA.

[Ahn et al. 2007] Ahn, Y.-Y., Han, S., Kwak, H., Moon, S., and Jeong, H. (2007). Analysis of topological characteristics of huge online social networking services. In *Proc. of the Int. Conf. on World Wide Web – WWW*, pages 835–844, Banff, Alberta, Canada.

[Ahsan et al. 2015] Ahsan, S., Bajpai, V., Ott, J., and Schönwälder, J. (2015). *Passive and Active Measurement: 16th International Conference, PAM 2015, New York, NY, USA, March 19-20, 2015, Proceedings*, chapter Measuring YouTube from Dual-Stacked Hosts, pages 249–261. Springer International Publishing, Cham.

[Akamai Technologies 2014] Akamai Technologies (2014). Akamai: State-of-the-internet-reports. `http://www.akamai.com/stateoftheinternet/`. [Último acesso: 01/04/2016].

[Al-Hazmi and Magedanz 2012] Al-Hazmi, Y. and Magedanz, T. (2012). A flexible monitoring system for federated future internet testbeds. In *Network of the Future (NOF), 2012 Third International Conference on the*, pages 1–6.

[ANATEL 2015] ANATEL (2015). Banda Larga - Acessos. `http://www.anatel.gov.br/dados/index.php?option=com_content&view=article&id=269:testeeee&catid=84&Itemid=506`. [Último acesso: 01/04/2016].

[Antichi et al. 2014] Antichi, G., Donatini, L., Garroppo, R. G., Giordano, S., and Moore, A. W. (2014). *Mathematical and Engineering Methods in Computer Science: 9th International Doctoral Workshop, MEMICS 2014, Telč, Czech Republic, October 17–19, 2014, Revised Selected Papers*, chapter An Open-Source Hardware Approach for High Performance Low-Cost QoS Monitoring of VoIP Traffic, pages 1–15. Springer International Publishing, Cham.

[Antoniades and Dovrolis 2015] Antoniades, D. and Dovrolis, C. (2015). Co-evolutionary dynamics in social networks: a case study of Twitter. *Computational Social Networks*, 2(1):1–21.

[Arlos et al. 2005] Arlos, P., Fiedler, M., and Nilsson, A. A. (2005). A distributed passive measurement infrastructure. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 215–227, Boston, MA, USA.

[Astuto et al. 2014] Astuto, B., Nunes, A., Mendonca, M., Nguyen, X.-N., Obraczka, K., and Turletti, T. (2014). A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. *IEEE Communications Surveys & Tutorials*, 16(3):1617–1634.

[Augustin et al. 2006] Augustin, B., Cuvellier, X., Orgogozo, B., Viger, F., Friedman, T., Latapy, M., Magnien, C., and Teixeira, R. (2006). Avoiding traceroute anomalies with paris traceroute. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 153–158, Rio de Janeriro, Brazil.

[Baltrunas et al. 2014] Baltrunas, D., Elmokashfi, A., and Kvalbein, A. (2014). Measuring the reliability of mobile broadband networks. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 45–58, Vancouver, BC, Canada.

[Barford and Sommers 2004] Barford, P. and Sommers, J. (2004). Comparing probe-based and router-based packet-loss measurement. *IEEE Internet Computing*, 8(5):50–56.

[Bavier et al. 2004] Bavier, A., Bowman, M., Chun, B., Culler, D., Karlin, S., Muir, S., Peterson, L., Roscoe, T., Spalink, T., and Wawrzoniak, M. (2004). Operating system support for planetary-scale network services. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, pages 19–19, Berkeley, CA, USA. USENIX Association.

[Becker et al. 2013] Becker, R., Cáceres, R., Hanson, K., Isaacman, S., Loh, J. M., Martonosi, M., Rowland, J., Urbanek, S., Varshavsky, A., and Volinsky, C. (2013). Human mobility characterization from cellular network data. *Communications of the ACM*, 56(1):74–82.

[Benevenuto et al. 2009] Benevenuto, F., Rodrigues, T., Cha, M., and Almeida, V. (2009). Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 49–62, New York, NY, USA. ACM.

[Bischof et al. 2014] Bischof, Z. S., Bustamante, F. E., and Stanojevic, R. (2014). Need, want, can afford – broadband markets and the behavior of users. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 73–86, Vancouver, BC, Canada.

[Blenk et al. 2016] Blenk, A., Basta, A., Reisslein, M., and Kellerer, W. (2016). Survey on network virtualization hypervisors for software defined networking. *IEEE Communications Surveys Tutorials*, 18(1):655–685.

[Bolot 1993] Bolot, J.-C. (1993). End-to-end packet delay and loss behavior in the Internet. In *Proc. of the ACM SIGCOMM*, pages 289–298, San Francisco, CA, USA.

[Bonfiglio et al. 2008] Bonfiglio, D., Mellia, M., Meo, M., Ritacca, N., and Rossi, D. (2008). Tracking down skype traffic. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*.

[Bourgeau et al. 2011] Bourgeau, T., Augé, J., and Friedman, T. (2011). *Testbeds and Research Infrastructures. Development of Networks and Communities: 6th International ICST Conference, TridentCom 2010, Berlin, Germany, May 18-20, 2010, Revised Selected Papers*, chapter TopHat: Supporting Experiments through Measurement Infrastructure Federation, pages 542–557. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Brito et al. 2015] Brito, S. H. B., Santos, M. A. S., dos Reis Fontes, R., Perez, D. A. L., and Rothenberg, C. E. (2015). Anatomia do ecossistema de pontos de troca de tráfego publicos na internet do brasil. *XXXIII Simpósio Brasileiro de Redes de Computadores (SBRC)*.

[Caizzone et al. 2008] Caizzone, G., Corghi, A., Giacomazzi, P., and Nonnoi, M. (2008). Analysis of the scalability of the overlay Skype system. In *Proc. of the IEEE Int. Conf. on Communications – ICC*, pages 5652–5658, Beijing, China.

[Calabrese et al. 2014] Calabrese, F., Ferrari, L., and Blondel, V. D. (2014). Urban sensing using mobile phone network data: A survey of research. *ACM Computing Surveys*, 47(2):25:1–25:20.

[Calvert et al. 2011] Calvert, K. L., Edwards, W. K., Feamster, N., Grinter, R. E., Deng, Y., and Zhou, X. (2011). Instrumenting home networks. *SIGCOMM Comput. Commun. Rev.*, 41(1):84–89.

[Campista et al. 2012] Campista, M. E. M., Amorim, M. D., and Costa, L. H. M. K. (2012). Big wireless measurement campaigns: Are they really worth the price? In *Proc. of the ACM International Workshop on Hot Topics in Planet-Scale Measurement – HotPlanet*, pages 27–32, Low Wood Bay, Lake District, UK.

[Casas et al. 2014] Casas, P., D'Alconzo, A., Fiadino, P., Bar, A., Finamore, A., and Zseby, T. (2014). When youtube does not work—analysis of qoe-relevant degradation in google cdn traffic. *Network and Service Management, IEEE Transactions on*, 11(4):441–457.

[Centola 2010] Centola, D. (2010). The spread of behavior in an online social network experiment. *Science*, 329(5996):1194–1197.

[Cha et al. 2009] Cha, M., Mislove, A., and Gummadi, K. P. (2009). A measurement-driven analysis of information propagation in the Flickr social network. In *Proc. of the Int. Conf. on World Wide Web – WWW*, WWW '09, pages 721–730, Madrid, Spain.

[Chen et al. 2015] Chen, F., Sitaraman, R. K., and Torres, M. (2015). End-user mapping: Next generation request routing for content delivery. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 167–181. ACM.

[Chen 2001] Chen, T. M. (2001). Increasing the observability of Internet behavior. *Communications of the ACM*, 44(1):93–98.

[Chen et al. 2003] Chen, Y., Bindel, D., and Katz, R. H. (2003). Tomography-based overlay network monitoring. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 216–231, Miami Beach, FL, USA.

[Chen et al. 2014] Chen, Y.-C., Liao, Y., Baldi, M., Lee, S.-J., and Qiu, L. (2014). OS fingerprinting and tethering detection in mobile networks. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 173–179, Vancouver, BC, Canada.

[Chen et al. 2013] Chen, Y.-C., sup Lim, Y., Gibbens, R. J., Nahum, E. M., Khalili, R., and Towsley, D. (2013). A measurement-based study of multipath TCP performance over wireless networks. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 455–468, Barcelona, Spain.

[Cheng et al. 2008] Cheng, X., Dale, C., and Liu, J. (2008). Statistics and social network of YouTube videos. In *Int. Workshop on Quality of Service – IWQoS*, pages 229–238, Enschede, The Netherlands.

[Chowdhury et al. 2014] Chowdhury, S. R., Ahmed, R., and Boutaba, R. (2014). PayLess: A Low Cost Network Monitoring Framework for Software Defined Networks. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 1–9. IEEE.

[Cisco Systems, Inc. 2015] Cisco Systems, Inc. (2015). Cisco visual networking index: Forecast and methodology. `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html`. [Último acesso: 01/04/2016].

[Ciullo et al. 2010] Ciullo, D., Garcia, M. A., Horvath, A., Leonardi, E., Mellia, M., Rossi, D., Telek, M., and Veglia, P. (2010). Network awareness of p2p live streaming applications: A measurement study. *IEEE Transactions on Multimedia*, 12(1):54–63.

[Clark et al. 2005] Clark, D., Wroclawski, J., Sollins, K., and Braden, R. (2005). Tussle in cyberspace: defining tomorrow's Internet. *IEEE/ACM Transactions on Networking*, 13(3):462–475.

[Crovella and Krishnamurthy 2006] Crovella, M. and Krishnamurthy, B. (2006). *Internet Measurement: Infrastructure, Traffic and Applications*. John Wiley and Sons, Inc.

[Crovella and Bestavros 1997] Crovella, M. E. and Bestavros, A. (1997). Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846.

[Cunha et al. 2009] Cunha, I., Teixeira, R., Feamster, N., and Diot, C. (2009). Measurement methods for fast and accurate blackhole identification with binary tomography. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 254–266, Chicago, IL, USA. ACM.

[da Silva and Rocha 2015] da Silva, D. and Rocha, A. (2015). Detecção de streamers em redes BitTorrent. In *XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, Vitória, ES.

[del Río et al. 2011] del Río, P. M. S., Ramos, J., García-Dorado, J. L., Aracil, J., Cuadra-Sánchez, A., and Cutanda-Rodríguez, M. (2011). On the processing time for detection of skype traffic. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, pages 1784–1788.

[Deng et al. 2014] Deng, S., Netravali, R., Sivaraman, A., and Balakrishnan, H. (2014). WiFi, LTE, or both? measuring multi-homed wireless Internet performance. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 181–194, Vancouver, BC, Canada.

[des Roziers et al. 2011] des Roziers, C. B., Chelius, G., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N., and Noël, T. (2011). SensLAB – Very large scale open wireless sensor network testbed. In *Proc. of the TRIDENTCOM*, pages 239–254, Shanghai, China.

[DiCioccio et al. 2012] DiCioccio, L., Teixeira, R., May, M., and Kreibich, C. (2012). Probe and pray: Using UPnP for home network measurements. In *Proc. of the Int. Conf. on Passive and Active Measurement (PAM)*, pages 96–105, Vienna, Austria.

[DiCioccio et al. 2013] DiCioccio, L., Teixeira, R., and Rosenberg, C. (2013). Measuring home networks with HomeNet profiler. In *Proc. of the Int. Conf. on Passive and Active Measurement (PAM)*, pages 176–186, Hong Kong, China.

[Dong et al. 2013] Dong, W., Duffield, N., Ge, Z., Lee, S., and Pang, J. (2013). Modeling cellular user mobility using a leap graph. In *Proc. of the Int. Conf. on Passive and Active Measurement (PAM)*, pages 53–62, Hong Kong, China.

[Donnet et al. 2005] Donnet, B., Raoult, P., Friedman, T., and Crovella, M. (2005). Efficient algorithms for large-scale topology discovery. *SIGMETRICS Performance Evaluation Review*, 33(1):327–338.

[Drago et al. 2013] Drago, I., Bocchi, E., Mellia, M., Slatman, H., and Pras, A. (2013). Benchmarking personal cloud storage. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 205–212, New York, NY, USA. ACM.

[Drago et al. 2012] Drago, I., Mellia, M., M. Munafo, M., Sperotto, A., Sadre, R., and Pras, A. (2012). Inside dropbox: Understanding personal cloud storage services. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 481–494, New York, NY, USA. ACM.

[Dunn et al. 2012] Dunn, C. W., Gupta, M., Gerber, A., and Spatscheck, O. (2012). Navigation characteristics of online social networks and search engines users. In *Proceedings of the 2012 ACM Workshop on Workshop on Online Social Networks*, WOSN '12, pages 43–48, New York, NY, USA. ACM.

[Eagle et al. 2009] Eagle, N., Pentland, A., and Lazer, D. (2009). Inferring social network structure using mobile phone data. *Proceedings of the National Academy of Sciences (PNAS)*, 106(36):15274–15278.

[Entidade Aferidora da Qualidade de Banda Larga 2015] Entidade Aferidora da Qualidade de Banda Larga (2015). Qualidade Serviço de Comunicação Multimídia. `http://www.teleco.com.br/qscm_qualidade.asp`. [Último acesso: 01/04/2016].

[Eriksson et al. 2012] Eriksson, B., Dasarathy, G., Barford, P., and Nowak, R. (2012). Efficient network tomography for Internet topology discovery. *IEEE/ACM Transactions on Networking*, 20(3):931–943.

[Fallica et al. 2008] Fallica, B., Lu, Y., Kuipers, F., Kooij, R., and Mieghem, P. V. (2008). On the quality of experience of sopcast. In *Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST '08. The Second International Conference on*, pages 501–506.

[Faloutsos et al. 1999] Faloutsos, M., Faloutsos, P., and Faloutsos, C. (1999). On power-law relationships of the Internet topology. In *Proc. of the ACM SIGCOMM*, pages 251–262, Cambridge, MA, USA.

[Fan et al. 2015] Fan, X., Katz-Bassett, E., and Heidemann, J. (2015). Assessing affinity between users and cdn sites. In *Traffic Monitoring and Analysis*, pages 95–110. Springer.

[Fattaholmanan and Rabiee 2015] Fattaholmanan, A. and Rabiee, H. (2015). A large-scale active measurement study on the effectiveness of piece-attack on bittorrent networks. *IEEE Transactions on Dependable and Secure Computing*, PP(99):1–1.

[Federal Communications Commission 2015] Federal Communications Commission (2015). 2015 Broadband Progress Report. `http://www.fcc.gov/document/fcc-finds-us-broadband-deployment-not-keeping-pace`. [Último acesso: 01/04/2016].

[Ferreira et al. 2016] Ferreira, F. H., da Silva, A. P. C., and Vieira, A. B. (2016). Characterizing peers communities and dynamics in a P2P live streaming system. *Peer-to-Peer Networking and Applications*, 9(1):1–15.

[Fleury et al. 2015] Fleury, E., Mitton, N., Noël, T., and Adjih, C. (2015). FIT IoT-LAB: The largest IoT open experimental testbed. *ERCIM News*, page 4.

[Floyd and Paxson 2001] Floyd, S. and Paxson, V. (2001). Difficulties in simulating the Internet. *IEEE/ACM Transactions on Networking*, 9(4):392–403.

[Freire et al. 2008] Freire, E. P., Ziviani, A., and Salles, R. M. (2008). Detecting voip calls hidden in web traffic. *IEEE Transactions on Network and Service Management*, 5(4):204–214.

[Fukuda et al. 2015] Fukuda, K., Asai, H., and Nagami, K. (2015). Tracking the evolution and diversity in network usage of smartphones. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 253–266, Tokyo, Japan.

[Ghimire et al. 2015] Ghimire, J., Mani, M., Crespi, N., and Sanguankotchakorn, T. (2015). Delay and capacity analysis of structured P2P overlay for lookup service. *Telecommunication Systems*, 58(1):33–54.

[Ghita et al. 2013] Ghita, D., Argyraki, K., and Thiran, P. (2013). Toward accurate and practical network tomography. *ACM SIGOPS Operating Systems Review*, 47(1):22–26.

[Goel et al. 2016] Goel, U., Wittie, M. P., Claffy, K. C., and Le, A. (2016). Survey of end-to-end mobile network measurement testbeds, tools, and services. *IEEE Communications Surveys & Tutorials*, 18(1):105–123.

[Goga and Teixeira 2012] Goga, O. and Teixeira, R. (2012). Speed measurements of residential internet access. In *Proc. of the Int. Conf. on Passive and Active Measurement (PAM)*, pages 168–178, Vienna, Austria.

[Gomes et al. 2013] Gomes, J. V., Inacio, P. R., Pereira, M., Freire, M. M., and Monteiro, P. P. (2013). Identification of peer-to-peer voip sessions using entropy and codec properties. *IEEE Transactions on Parallel and Distributed Systems*, 24(10):2004–2014.

[González et al. 2008] González, M. C., Hidalgo, C. A., and Barabási, A.-L. (2008). Understanding individual human mobility patterns. *Nature*, 453.

[Gracia-Tinedo et al. 2013] Gracia-Tinedo, R., Artigas, M. S., Moreno-Martínez, A., Cotes, C., and López, P. G. (2013). Actively measuring personal cloud storage. In *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*, pages 301–308.

[Gracia-Tinedo et al. 2015] Gracia-Tinedo, R., Tian, Y., Sampé, J., Harkous, H., Lenton, J., García-López, P., Sánchez-Artigas, M., and Vukolic, M. (2015). Dissecting ubuntuone: Autopsy of a global-scale personal cloud back-end. In *Proceedings of the 2015 ACM Conference on Internet Measurement Conference*, IMC '15, pages 155–168, New York, NY, USA. ACM.

[Grover et al. 2013] Grover, S., Park, M. S., Sundaresan, S., Burnett, S., Kim, H., Ravi, B., and Feamster, N. (2013). Peeking behind the nat: An empirical study of home networks. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 377–390, New York, NY, USA. ACM.

[Guille et al. 2013] Guille, A., Hacid, H., Favre, C., and Zighed, D. A. (2013). Information diffusion in online social networks: A survey. *SIGMOD Record*, 42(2):17–28.

[Gummadi et al. 2003] Gummadi, K. P., Dunn, R. J., Saroiu, S., Gribble, S. D., Levy, H. M., and Zahorjan, J. (2003). Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. *SIGOPS Oper. Syst. Rev.*, 37(5):314–329.

[Gunduz and Yuksel 2016] Gunduz, G. and Yuksel, M. (2016). Popularity-based scalable peer-to-peer topology growth. *Computer Networks*, 100:124 – 140.

[Guo et al. 2005] Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., and Zhang, X. (2005). Measurements, analysis, and modeling of BitTorrent-like systems. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 35–48, Berkeley, CA, USA.

[Habib et al. 2004] Habib, A., Khan, M., and Bhargava, B. (2004). Edge-to-edge measurement-based distributed network monitoring. *Computer Networks*, 44(2):211–233.

[Hanemann et al. 2005] Hanemann, A., Boote, J. W., Boyd, E. L., Durand, J., Kudarimoti, L., apacz, R., Swany, D. M., Trocha, S., and Zurawski, J. (2005). PerfSONAR: A service oriented architecture for multi-domain network monitoring. In *Int. Conf. on Service-Oriented Computing (ICSOC)*, volume 3826 of *Lecture Notes in Computer Science*, pages 241–254.

[He et al. 2015] He, T., Liu, C., Swami, A., Towsley, D., Salonidis, T., Bejan, A. I., and Yu, P. (2015). Fisher information-based experiment design for network tomography. *SIGMETRICS Performance Evaluation Review*, 43(1):389–402.

[Hei et al. 2007] Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. W. (2007). A measurement study of a large-scale P2P IPTV system. *IEEE Transactions on Multimedia*, 9(8):1672–1687.

[Heidemann et al. 2012] Heidemann, J., Klier, M., and Probst, F. (2012). Online social networks: A survey of a global phenomenon. *Computer Networks*, 56(18):3866–3878.

[Hendriks et al. 2016] Hendriks, L., de O. Schmidt, R., Sadre, R., Bezerra, J. A., and Pras, A. (2016). Assessing the quality of flow measurements from openflow devices. In *8th International Workshop on Traffic Monitoring and Analysis (TMA)*.

[Hess et al. 2015] Hess, A., Hummel, K. A., Gansterer, W. N., and Haring, G. (2015). Data-driven human mobility modeling: A survey and engineering guidance for mobile networking. *ACM Computing Surveys*, 48(3):38:1–38:39.

[Hu et al. 2015] Hu, X., Chu, T. H. S., Leung, V. C. M., Ngai, E. C. H., Kruchten, P., and Chan, H. C. B. (2015). A survey on mobile social networks: Applications, platforms, system architectures, and future research directions. *IEEE Communications Surveys Tutorials*, 17(3):1557–1581.

[Huberman and Adamic 1999] Huberman, B. A. and Adamic, L. A. (1999). Internet: Growth dynamics of the World-Wide Web. *Nature*, 401(6749):131.

[Imbrenda et al. 2014] Imbrenda, C., Muscariello, L., and Rossi, D. (2014). Analyzing cacheable traffic in isp access networks for micro cdn applications via content-centric networking. In *Proceedings of the 1st international conference on Information-centric networking*, pages 57–66. ACM.

[Ishibashi et al. 2004] Ishibashi, K., Kanazawa, T., Aida, M., and Ishii, H. (2004). Active/passive combination-type performance measurement method using change-of-measure framework. *Computer Communications*, 27(9):868–879.

[Isolani et al. 2015] Isolani, P. H., Wickboldt, J. A., Both, C. B., Rochol, J., and Granville, L. Z. (2015). Interactive Monitoring, Visualization, and Configuration of OpenFlow-Based SDN. In *Proc. of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1–9.

[Jaynes 1957] Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical review*, 106(4):620.

[Jiang et al. 2013] Jiang, J., Wilson, C., Wang, X., Sha, W., Huang, P., Dai, Y., and Zhao, B. Y. (2013). Understanding latent interactions in online social networks. *ACM Transactios on the Web*, 7(4):18:1–18:39.

[Jin et al. 2013] Jin, L., Chen, Y., Wang, T., Hui, P., and Vasilakos, A. V. (2013). Understanding user behavior in online social networks: A survey. *IEEE Communications Magazine*, 51(9):144–150.

[Joe-Wong et al. 2015] Joe-Wong, C., Ha, S., Sen, S., , and Chiang, M. (2015). Do mobile data plans affect usage? results from a pricing trial with isp customers. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 96–108, New York, NY, USA.

[Kakhki et al. 2015] Kakhki, A. M., Razaghpanah, A., Li, A., Koo, H., Golani, R., Choffnes, D., Gill, P., and Mislove, A. (2015). Identifying traffic differentiation in mobile networks. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 239–251, Tokyo, Japan.

[Karagiannis et al. 2004] Karagiannis, T., Broido, A., Brownlee, N., Claffy, K. C., and Faloutsos, M. (2004). Is P2P dying or just hiding? In *Proc. of the IEEE GLOBECOM*, volume 3, pages 1532–1538 Vol.3.

[Karagiannis et al. 2005] Karagiannis, T., Papagiannaki, K., and Faloutsos, M. (2005). Blinc: Multilevel traffic classification in the dark. In *Proc. of the ACM SIGCOMM*, pages 229–240, Philadelphia, PA, USA.

[Kone et al. 2011] Kone, V., Zheleva, M., Wittie, M., Zhao, B. Y., Belding, E. M., Zheng, H., and Almeroth, K. C. (2011). AirLab: Consistency, fidelity and privacy in wireless measurements. *ACM SIGCOMM Computer Communication Review*, 41(1):60–65.

[Kreutz et al. 2014] Kreutz, D., Ramos, F. M. V., Veríssimo, P., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2014). Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, 103(1):63.

[Krishnappa et al. 2013] Krishnappa, D. K., Bhat, D., and Zink, M. (2013). Dashing youtube: An analysis of using dash in youtube video service. *39th Annual IEEE Conference on Local Computer Networks*, 0:407–415.

[Labovitz 2012] Labovitz, C. (2012). First data on changing netflix and cdn market share. http://www.deepfield.net/2012/06/first-data-onchanging- netflix-and-cdn-market-share/.

[Lakhina et al. 2004] Lakhina, A., Crovella, M., and Diot, C. (2004). Diagnosing network-wide traffic anomalies. In *Proc. of the ACM SIGCOMM*, pages 219–230, Portland, Oregon, USA.

[Leitão et al. 2012] Leitão, J., Marques, J. P., Pereira, J., and Rodrigues, L. (2012). X-BOT: A protocol for resilient optimization of unstructured overlay networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(11):2175–2188.

[Leland et al. 1993] Leland, W. E., Taqqu, M. S., Willinger, W., and Wilson, D. V. (1993). On the self-similar nature of ethernet traffic. In *Proc. of the ACM SIGCOMM*, pages 183–193, San Francisco, CA, USA.

[Liu et al. 2015] Liu, C., He, T., Swami, A., Towsley, D., Salonidis, T., Bejan, A. I., and Yu, P. (2015). Multicast vs. unicast for loss tomography on tree topologies. In *Proc. of the IEEE Military Communications Conference – MILCOM*, pages 312–317, Tampa, FL, USA.

[Liu et al. 2008] Liu, Y., Guo, Y., and Liang, C. (2008). A survey on peer-to-peer video streaming systems. *Peer-to-Peer Networking and Applications*, 1:18–28.

[Lochin 2010] Lochin, E. (2010). STAMP: SMTP server topological analysis by message headers parsing. In *Proc. of the IEEE Consumer Communications and Networking Conference – CCNC*, pages 1–2, Las Vegas, NV, USA.

[Lodhi et al. 2014] Lodhi, A., Larson, N., Dhamdhere, A., Dovrolis, C., et al. (2014). Using peeringdb to understand the peering ecosystem. *ACM SIGCOMM Computer Communication Review*, 44(2):20–27.

[Lopez et al. 2014] Lopez, P. G., Sanchez-Artigas, M., Toda, S., Cotes, C., and Lenton, J. (2014). Stacksync: Bringing elasticity to dropbox-like file synchronization. In *Proceedings of the 15th International Middleware Conference*, Middleware '14, pages 49–60, New York, NY, USA. ACM.

[Lua et al. 2005] Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., and Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys & Tutorials*, 7(2):72–93.

[Ma et al. 2014] Ma, L., He, T., Leung, K. K., Swami, A., and Towsley, D. (2014). Monitor placement for maximal identifiability in network tomography. In *Proc. of the IEEE INFOCOM*, pages 1447–1455.

[Madruga et al. 2015] Madruga, E. L., David, R., de Souza, R. S., and Dantas, R. (2015). Assessing quality of experience while comparing competing mobile broadband services from the user perspective. *Journal of Physics: Conference Series*, 575(012012).

[Mahadevan et al. 2006] Mahadevan, P., Krioukov, D., Fomenkov, M., Dimitropoulos, X., claffy, k. c., and Vahdat, A. (2006). The Internet AS-level topology: Three data sources and one definitive metric. *ACM SIGCOMM Computer Commuhication Review*, 36(1):17–26.

[Malatras 2015] Malatras, A. (2015). State-of-the-art survey on {P2P} overlay networks in pervasive computing environments. *Journal of Network and Computer Applications*, 55:1 – 23.

[Marcondes et al. 2012] Marcondes, C., Martins, J., Suruagy, J., Cardoso, K., Abelém, A., Nascimento, V., Machado, I., Carvalho, T., Miers, C., Salvador, M., and Rothenberg, C. E. (2012). Estado da Arte de Sistemas de Controle e Monitoramento de Infraestruturas para Experimentação de Redes de Comunicação. In *Minicursos do SBRC*. SBC, Ouro Preto, MG.

[Mendoza et al. 2015] Mendoza, A., Singh, K., and Gu, G. (2015). What is wrecking your data plan? a measurement study of mobile web overhead. In *Proc. of INFOCOM*, pages 2740–2748, Hong Kong, China.

[Miniwatts Marketing Group 2016] Miniwatts Marketing Group (2016). Internet World Stats. `http://www.internetworldstats.com/`. [Último acesso: 01/04/2016].

[Mislove et al. 2007] Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P., and Bhattacharjee, B. (2007). Measurement and analysis of online social networks. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 29–42, San Diego, CA, USA.

[Moraes et al. 2014] Moraes, I. M., Mattos, D. M., Ferraz, L. H. G., Campista, M. E. M., Rubinstein, M. G., Costa, L. H. M., de Amorim, M. D., Velloso, P. B., Duarte, O. C. M., and Pujolle, G. (2014). Fits: A flexible virtual network testbed architecture. *Computer Networks*, 63:221 – 237. Special issue on Future Internet Testbeds ? Part {II}.

[Motamedi et al. 2015] Motamedi, R., Rejaie, R., and Willinger, W. (2015). A survey of techniques for Internet topology discovery. *IEEE Communications Surveys & Tutorials*, 17(2):1044–1065.

[Naboulsi et al. 2014] Naboulsi, D., Stanica, R., and Fiore, M. (2014). Classifying call profiles in large-scale mobile traffic datasets. In *Proc. of INFOCOM*, pages 1806–1814, Toronto, ON, Canada.

[Nakao et al. 2006] Nakao, A., Peterson, L., and Bavier, A. (2006). Scalable routing overlay networks. *SIGOPS Operating Systems Review*, 40(1):49–61.

[Nikravesh et al. 2014] Nikravesh, A., Choffnes, D. R., Katz-Bassett, E., Mao, Z. M., and Welsh, M. (2014). Mobile network performance from user devices: A longitudinal, multidimensional analysis. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 12–22, Los Angeles, CA, USA.

[Norton 2011] Norton, W. B. (2011). *The Internet peering playbook: connecting to the core of the Internet*. DrPeering Press.

[Noulas et al. 2013] Noulas, A., Mascolo, C., and Frias-Martinez, E. (2013). Exploiting foursquare and cellular data to infer user activity in urban environments. In *Proc. of the IEEE Int. Conf. on Mobile Data Management (MDM)*, pages 167–176, Milan, Italy.

[Nucci and Papagiannnaki 2009] Nucci, A. and Papagiannnaki, K. (2009). *Design, Measurement and Management of Large-Scale IP Networks*. Cambridge University Press.

[ONF 2015] ONF (2015). OpenFlow Switch Specification Version 1.5.1. Technical report, The Open Networking Foundation.

[Pallis and Vakali 2006] Pallis, G. and Vakali, A. (2006). Insight and perspectives for content delivery networks. *Communications of the ACM*, 49(1):101–106.

[Papagelis et al. 2013] Papagelis, M., Das, G., and Koudas, N. (2013). Sampling online social networks. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):662–676.

[Paxson 1997] Paxson, V. (1997). End-to-end Internet packet dynamics. In *Proc. of the ACM SIGCOMM*, pages 139–152, Cannes, France.

[Paxson and Floyd 1995] Paxson, V. and Floyd, S. (1995). Wide area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244.

[Pefkianakis et al. 2015] Pefkianakis, I., Lundgren, H., Soule, A., Chandrashekar, J., Guyadec, P. L., Diot, C., May, M., Doorselaer, K. V., and Oost, K. V. (2015). Characterizing home wireless performance: The gateway view. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 2713–2731.

[Perenyi et al. 2007] Perenyi, M., Gefferth, A., Dang, T. D., and Molnar, S. (2007). Skype traffic identification. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 399–404.

[Pinheiro et al. 2012] Pinheiro, M. M., Macêdo, I. L. E., Souza, I. L. O., Hohlenweger, T. S., Leite, P. R. R., Spínola, A. L., Monteiro, H., Dourado, R. A., Sampaio, L. N., Monteiro, J. A. S., and Martins, J. S. B. (2012). *Testbeds and Research Infrastructure. Development of Networks and Communities: 8th International ICST Conference, TridentCom 2012, Thessanoliki, Greece, June 11-13, 2012, Revised Selected Papers*, chapter An Instrumentation and Measurement Architecture Supporting Multiple Control Monitoring Frameworks, pages 363–364. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Ponieman et al. 2013] Ponieman, N. B., Salles, A., and Sarraute, C. (2013). Human mobility and predictability enriched by social phenomena information. In *Proc. of the IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining – ASONAM*, pages 1331–1336, Niagara, Ontario, Canada.

[Pouwelse et al. 2005] Pouwelse, J., Garbacki, P., Epema, D., and Sips, H. (2005). *Peer-to-Peer Systems IV: 4th International Workshop, IPTPS 2005, Ithaca, NY, USA, February 24-25, 2005. Revised Selected Papers*, chapter The Bittorrent P2P File-Sharing System: Measurements and Analysis, pages 205–216. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Prasad et al. 2003] Prasad, R., Dovrolis, C., Murray, M., and Claffy, K. (2003). Bandwidth estimation: metrics, measurement techniques, and tools. *IEEE Network*, 17(6):27–35.

[Qin et al. 2014] Qin, P., Dai, B., Huang, B., Xu, G., and Wu, K. (2014). A survey on network tomography with network coding. *IEEE Communications Surveys Tutorials*, 16(4):1981–1995.

[Qiu and Srikant 2004] Qiu, D. and Srikant, R. (2004). Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In *Proc. of the ACM SIGCOMM*, pages 367–378, Portland, OR, USA.

[Raghuramu et al. 2015] Raghuramu, A., Zang, H., and Chuah, C.-N. (2015). Uncovering the footprints of malicious traffic in cellular data networks. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 70–82, New York, NY, USA.

[Rakotoarivelo et al. 2010] Rakotoarivelo, T., Ott, M., Jourjon, G., and Seskar, I. (2010). Omf: A control and management framework for networking testbeds. *SIGOPS Oper. Syst. Rev.*, 43(4):54–59.

[Raychaudhuri et al. 2005] Raychaudhuri, D., Seskar, I., Ott, M., Ganu, S., Ramachandran, K., Kremo, H., Siracusa, R., Liu, H., and Singh, M. (2005). Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pages 1664–1669 Vol. 3.

[Richter et al. 2015] Richter, P., Chatzis, N., Smaragdakis, G., Feldmann, A., and Willinger, W. (2015). Distilling the internet's application mix from packet-sampled traffic. In *Passive and Active Measurement*, pages 179–192. Springer.

[Rotsos et al. 2012] Rotsos, C., Sarrar, N., Uhlig, S., Sherwood, R., and Moore, A. W. (2012). Oflops: An open framework for openflow switch evaluation. In *Proceedings of the 13th International Conference on Passive and Active Measurement*, PAM'12, pages 85–95, Berlin, Heidelberg. Springer-Verlag.

[Sampaio et al. 2015] Sampaio, L. N., de Oliveira, L. R., Barreto, M. E., and da Silva Bezerra end Allan Edgard Silva Freitas, R. M. (2015). Bambu: A metropolitan innovation testbed for promoting future internet research. In *SwitchOn - Research, Collaboration and Education*.

[Sampaio et al. 2007] Sampaio, L. N., Koga, I., Costa, R., Monteiro, H., Monteiro, J. A. S., Vetter, F., Fernandes, G., and Vetter, M. (2007). Implementing and deploying network monitoring service oriented architectures. In *LANOMS*, pages 28–37.

[Schaffers 2015] Schaffers, H. (2015). Future internet research and experimentation: Vision, strategy and roadmap towards 2020. Technical Report 1, AmpliFIRE Coordination and Support Action.

[Schneider et al. 2009] Schneider, F., Feldmann, A., Krishnamurthy, B., and Willinger, W. (2009). Understanding online social network usage from a network perspective. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 35–48, New York, NY, USA. ACM.

[Sengul et al. 2012] Sengul, C., Viana, A. C., and Ziviani, A. (2012). A survey of adaptive services to cope with dynamics in wireless self-organizing networks. *ACM Computing Surveys*, 44(4):23:1–23:35.

[Shafiq et al. 2014] Shafiq, M. Z., Liu, A. X., and Khakpour, A. R. (2014). Revisiting caching in content delivery networks. *ACM SIGMETRICS Performance Evaluation Review*, 42(1):567–568.

[Siekkinen et al. 2007] Siekkinen, M., Collange, D., Urvoy-Keller, G., and Biersack, E. W. (2007). Performance limitations of adsl users: A case study. In *Passive and Active Network Measurement*, pages 145–154. Springer.

[Siganos et al. 2009] Siganos, G., Pujol, J. M., and Rodriguez, P. (2009). Monitoring the bittorrent monitors: A bird's eye view. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 175–184, Berlin, Heidelberg. Springer-Verlag.

[Silveira et al. 2016] Silveira, L. M., Almeida, J. M., Marques-Neto, H. T., Sarraute, C., and Ziviani, A. (2016). MobHet: Predicting human mobility using heterogeneous data sources. Em submissão.

[Silveira et al. 2015] Silveira, L. M., Almeida, J. M., Marques-Neto, H. T., and Ziviani, A. (2015). MobDatU: um novo modelo de previsão de mobilidade humana para dados heterogêneos. In *Anais do Simp. Bras. de Redes de Computadores e Sistemas Distribuídos – SBRC*, pages 217–227, Vitória, ES.

[Silverston et al. 2009] Silverston, T., Fourmaux, O., Botta, A., Dainotti, A., Pescapé, A., Ventre, G., and Salamatian, K. (2009). Traffic analysis of peer-to-peer {IPTV} communities. *Computer Networks*, 53(4):470 – 484. Content Distribution Infrastructures for Community Networks.

[Singh et al. 2005] Singh, M., Ottand, and Kamat, P. (2005). ORBIT Measurements Framework and Library (OML): Motivations, Design,Implementation, and Features. In *Proceedings of the IEEE Tridentcom 2005*, Trento, Italy.

[Soule et al. 2005] Soule, A., Lakhina, A., Taft, N., Papagiannaki, K., Salamatian, K., Nucci, A., Crovella, M., and Diot, C. (2005). Traffic matrices: Balancing measurements, inference and modeling. In *Proc. of the ACM SIGMETRICS*, Banff, Canada.

[Spring et al. 2002] Spring, N., Mahajan, R., and Wetherall, D. (2002). Measuring ISP topologies with Rocketfuel. In *Proc. of the ACM SIGCOMM*, pages 133–145, Pittsburgh, PA, USA.

[Spring et al. 2003] Spring, N., Wetherall, D., and Anderson, T. (2003). Scriptroute: A public Internet measurement facility. In *Proc. of the USENIX Symp. on Internet Technologies and Systems – USITS*, pages 17–17, Seattle, WA.

[Stutzbach and Rejaie 2005] Stutzbach, D. and Rejaie, R. (2005). Characterizing the two-tier Gnutella topology. In *Proc. of the ACM SIGMETRICS*, pages 402–403, Banff, Canada.

[Stutzbach and Rejaie 2006] Stutzbach, D. and Rejaie, R. (2006). Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 189–202, New York, NY, USA. ACM.

[Suñé et al. 2014] Suñé, M., Bergesio, L., Woesner, H., Rothe, T., Köpsel, A., Colle, D., Puype, B., Simeonidou, D., Nejabati, R., Channegowda, M., Kind, M., Dietz, T., Autenrieth, A., Kotronis, V., Salvadori, E., Salsano, S., Körner, M., and Sharma, S. (2014). Design and implementation of the ofelia fp7 facility: The european openflow testbed. *Comput. Netw.*, 61:132–150.

[Sundaresan et al. 2011] Sundaresan, S., de Donato, W., Feamster, N., Teixeira, R., Crawford, S., and Pescapè, A. (2011). Broadband Internet Performance: A View from the Gateway. In *Proc. of the ACM SIGCOMM*, pages 134–145, Toronto, ON, Canada.

[Sundaresan et al. 2015] Sundaresan, S., Feamster, N., and Teixeira, R. (2015). Measuring the Performance of User Traffic in Home Wireless Networks. In *Proc. of the Int. Conf. on Passive and Active Measurement (PAM)*, pages 305–317, New York, NY, USA.

[Sundaresan et al. 2016] Sundaresan, S., Feamster, N., and Teixeira, R. (2016). Home network or access link? locating last-mile downstream throughput bottlenecks. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 111–123, Heraklion, Greece.

[Tang et al. 2015] Tang, H., Liu, F., Shen, G., Jin, Y., and Guo, C. (2015). Unidrive: Synergize multiple consumer cloud storage services. In *Proceedings of the 16th Annual Middleware Conference*, Middleware '15, pages 137–148, New York, NY, USA. ACM.

[Telecommunication Development Sector 2015] Telecommunication Development Sector (2015). Internet System Consortiun. `http://www.itu.int/en/ITU-D/Statistics/`. [Último acesso: 01/04/2016].

[Toole et al. 2015] Toole, J. L., Colak, S., Sturt, B., Alexander, L. P., Evsukoff, A., and González, M. C. (2015). The path most traveled: Travel demand estimation using big data resources. *Transportation Research Part C: Emerging Technologies*, 58, Part B:162–177. Big Data in Transportation and Traffic Engineering.

[Tootoonchian et al. 2012] Tootoonchian, A., Gorbunov, S., Ganjali, Y., Casado, M., and Sherwood, R. (2012). On controller performance in software-defined networks. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, Hot-ICE'12, pages 10–10, Berkeley, CA, USA. USENIX Association.

[Topirceanu et al. 2016] Topirceanu, A., Duma, A., and Udrescu, M. (2016). Uncovering the fingerprint of online social networks using a network motif based approach. *Computer Communications*, 73, Part B:167–175.

[Tozal and Sarac 2010] Tozal, M. E. and Sarac, K. (2010). TraceNET: An internet topology data collector. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 356–368, Melbourne, Australia.

[Traverso et al. 2015] Traverso, S., Abeni, L., Birke, R., Kiraly, C., Leonardi, E., Lo Cigno, R., and Mellia, M. (2015). Neighborhood filtering strategies for overlay construction in P2P-TV systems: Design and experimental comparison. *IEEE/ACM Transactions on Networking*, 23(3):741–754.

[Tune and Roughan 2015] Tune, P. and Roughan, M. (2015). Spatiotemporal traffic matrix synthesis. In *Proc. of the ACM SIGCOMM*, pages 579–592, London, United Kingdom.

[Tutschku 2004] Tutschku, K. (2004). *Passive and Active Network Measurement: 5th International Workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19-20, 2004. Proceedings*, chapter A Measurement-Based Traffic Profile of the eDonkey Filesharing Service, pages 12–21. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Ugander et al. 2011] Ugander, J., Karrer, B., Backstrom, L., and Marlow, C. (2011). The anatomy of the Facebook social graph. *CoRR*, abs/1111.4503.

[Van Adrichem et al. 2014] Van Adrichem, N. L. M., Doerr, C., and Kuipers, F. a. (2014). OpenNetMon: Network monitoring in OpenFlow software-defined networks. *IEEE/IFIP NOMS 2014 - IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World*.

[Wang et al. 2012] Wang, H., Shea, R., Wang, F., and Liu, J. (2012). On the impact of virtualization on dropbox-like cloud file storage/synchronization services. In *Quality of Service (IWQoS), 2012 IEEE 20th International Workshop on*, pages 1–9.

[Wang et al. 2015] Wang, H., Xu, F., Li, Y., Zhang, P., and Jin, D. (2015). Understanding mobile traffic patterns of large scale cellular towers in urban environment. In *Proc. of the ACM Internet Measurement Conference – IMC*, pages 225–238, Tokyo, Japan.

[Wang and Loguinov 2010] Wang, X. and Loguinov, D. (2010). Understanding and Modeling the Internet Topology: Economics and Evolution Perspective. *IEEE/ACM Transactions on Networking*, 18(1):257–270.

[Wattenhofer et al. 2012] Wattenhofer, M., Wattenhofer, R., and Zhu, Z., editors (2012). *The YouTube Social Network*, Dublin, Ireland.

[Wittie et al. 2010] Wittie, M. P., Pejovic, V., Deek, L., Almeroth, K. C., and Zhao, B. Y. (2010). Exploiting locality of interest in online social networks. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 25:1–25:12, New York, NY, USA. ACM.

[Wu et al. 2009] Wu, C. C., Chen, K. T., Chang, Y. C., and Lei, C. L. (2009). Peer-to-peer application recognition based on signaling activity. In *Communications, 2009. ICC 09. IEEE International Conference on*, pages 1–5.

[Xavier et al. 2013] Xavier, F. H. Z., Silveira, L. M., Almeida, J. M., Ziviani, A., Malab, C. H. S., and Marques-Neto, H. (2013). Understanding human mobility due to large-scale events. In *Proc. of the Int. Conf. on the Analysis of Mobile Phone Datasets – NetMob*, pages 45–47, Cambridge, MA, USA. MIT.

[Xavier et al. 2012] Xavier, F. H. Z., Silveira, L. M., Almeida, J. M., Ziviani, A., Malab, C. H. S., and Marques-Neto, H. T. (2012). Analyzing the workload dynamics of a mobile phone network in large scale events. In *Proc. of the Workshop on Urban Networking – UrbaNe, ACM CoNEXT*, pages 37–42, Nice, France.

[Xing et al. 2014] Xing, X., Meng, W., Doozan, D., Feamster, N., Lee, W., and Snoeren, A. C. (2014). Exposing inconsistent web search results with bobble. In *Proceedings of the 15th International Conference on Passive and Active Measurement - Volume 8362*, PAM 2014, pages 131–140, New York, NY, USA. Springer-Verlag New York, Inc.

[Xu et al. 2014] Xu, Y., Wang, Z., Leong, W. K., and Leong, B. (2014). An end-to-end measurement study of modern cellular data networks. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 34–45, Los Angeles, CA, USA.

[Yassine et al. 2015] Yassine, A., Rahimi, H., and Shirmohammadi, S. (2015). Software defined network traffic measurement: Current trends and challenges. *IEEE Instrumentation Measurement Magazine*, 18(2):42–50.

[Yeo et al. 2004] Yeo, C. K., Lee, B. S., and Er, M. H. (2004). A survey of application level multicast techniques. *Computer Communications*, 27(15):1547–1568.

[Yu et al. 2014] Yu, C., Xu, Y., Liu, B., and Liu, Y. (2014). "Can you SEE me now?" A measurement study of mobile video calls. In *Proc. of INFOCOM*, pages 1456–1464, Toronto, ON, Canada.

[Yu et al. 2013] Yu, M., Jose, L., and Miao, R. (2013). Software defined traffic measurement with opensketch. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 29–42, Berkeley, CA, USA. USENIX Association.

[Yu et al. 2006] Yu, Y., Liu, D., Li, J., and Shen, C. (2006). Traffic identification and overlay measurement of skype. In *Computational Intelligence and Security, 2006 International Conference on*, volume 2, pages 1043–1048.

[Yuan et al. 2014] Yuan, Z., Du, C., Chen, X., Wang, D., and Xue, Y. (2014). Skytracer: Towards fine-grained identification for skype traffic via sequence signatures. In *Computing, Networking and Communications (ICNC), 2014 International Conference on*, pages 1–5.

[Zenuni et al. 2014] Zenuni, X., Ajdari, J., Ismaili, F., and Raufi, B. (2014). Cloud storage providers: A comparison review and evaluation. In *Proceedings of the 15th International Conference on Computer Systems and Technologies*, CompSysTech '14, pages 272–277, New York, NY, USA. ACM.

[Zhang et al. 2005] Zhang, B., Liu, R., Massey, D., and Zhang, L. (2005). Collecting the internet as-level topology. *ACM SIGCOMM Computer Commuhication Review*, 35(1):53–61.

[Zhang et al. 2014] Zhang, D., Huang, J., Li, Y., Zhang, F., Xu, C., and He, T. (2014). Exploring human mobility with multi-source data at extremely large metropolitan scales. In *Proc. of the Int. Conf. on Mobile Computing and Networking – MOBICOM*, pages 201–212, Maui, Hawaii, USA. ACM.

[Zhang et al. 2015] Zhang, L., Xu, C., Pathak, P. H., and Mohapatra, P. (2015). Characterizing instant messaging apps on smartphones. In *Proc. of the Int. Conf. on Passsive and Active Measurements – PAM*, pages 83–95, New York, NY, USA.

[Zhou et al. 2007] Zhou, M., Dai, Y., and Li, X. (2007). A measurement study of the structured overlay network in p2p file-sharing systems. *Adv. MultiMedia*, 2007(1):10–10.

[Zink et al. 2009] Zink, M., Suh, K., Gu, Y., and Kurose, J. (2009). Characteristics of youtube network traffic at a campus network - measurements, models, and implications. *Comput. Netw.*, 53(4):501–514.

[Ziviani et al. 2012] Ziviani, A., Cardozo, T. B., and Gomes, A. T. A. (2012). Rapid prototyping of active measurement tools. *Computer Networks*, 56(2):870–883.

[Ziviani and Duarte 2005] Ziviani, A. and Duarte, O. C. M. B. (2005). Metrologia na Internet. In *Minicursos do SBRC*, pages 285–329. SBC, Fortaleza, CE.

Capítulo 5

Roteamento por Segmentos: Conceitos, Desafios e Aplicações Práticas

Antonio José Silvério (UFRJ/Embratel), Miguel Elias M. Campista (UFRJ) e Luís Henrique M. K. Costa (UFRJ)

Abstract

Currently, among the main problems of telecom operators are the huge number of states in the routers, manual configuration of traffic engineering and restoration paths in the network core. In this direction, the Segment Routing is an emerging proposal to simplify the routing and configuration of these networks. In Segment Routing, the flow states are kept only on the network edge nodes and configuration of the IP/MPLS core network, commonly used by operators, can be automated. To do this, Segment Routing uses the programming benefits of software defined network, by control plane centralization. The centralized controller view allows the dynamic calculation of the segments and the route construction between the edge nodes. Among the challenges to be explored in the short course are the problems faced by current operators and its solution through the Segment Routing.

Resumo

Atualmente, dentre os principais problemas das operadoras de telecomunicações estão o enorme número de estados nos roteadores, configurações manuais de engenharia de tráfego e restauração de caminhos no núcleo da rede. Nessa direção, o Roteamento por Segmentos é uma proposta emergente para simplificação do roteamento e da configuração dessas redes. No Roteamento por Segmentos, os estados por fluxo são mantidos apenas nos nós de borda da rede e a configuração das redes de núcleo IP/MPLS, comumente utilizadas pelas operadoras, pode ser automatizada. Para tal, o Roteamento por Segmentos utiliza os benefícios da programação das redes definidas por software através da centralização do plano de controle. A visão centralizada do controlador permite o cálculo dinâmico dos segmentos e a construção de rotas entre os nós de borda. Dentre os desafios a serem explorados no minicurso estão os problemas enfrentados pelas operadoras atuais e a solução através do Roteamento por Segmento.

5.1. Introdução

Na arquitetura TCP/IP o encaminhamento dos pacotes é feito salto a salto. Os campos do cabeçalho do pacote IP são analisados consultando-se a tabela de roteamento que contém os prefixos para as redes de destino. Quanto maior a tabela de roteamento, mais processamento a busca de rotas exige [De Ghein, 2007], aumentando a latência e jitter dos fluxos. Assim, um dos objetivos da tecnologia MPLS (*Multi Protocol Label Switiching*) foi lidar com o aumento do número de rotas nos roteadores de núcleo. Através do encaminhamento por rótulos de tamanho fixo, o MPLS possibilita acelerar a comutação dos pacotes [Santos et al., 2007].

As redes MPLS vem sendo adotadas como tecnologia típica da rede de roteadores de núcleo das operadora de telecomunicações. A tecnologia IP/MPLS provê um encaminhamento orientado a conexão através de circuitos virtuais unidirecionais. Inicialmente, tais circuitos eram estabelecidos seguindo os caminhos calculados pelo IGP (*Interior Gateway Protocol*) da rede, que tipicamente indicam o caminho mais curto. A inserção e remoção dos rótulos requer que os roteadores de borda conheçam o mapeamento entre rótulos MPLS e prefixos IP, configurados previamente.

Um problema que ocorre ao utilizar os caminhos calculados pelo IGP é a prevenção de situações de congestionamento ou do uso de caminhos de maior latência mesmo que com poucos saltos. Para contornar esse desafio, técnicas de engenharia de tráfego são frequentemente utilizadas para que os fluxos de dados utilizem rotas específicas por exemplo evitando congestionamento ou atendendo requisitos de qualidade de serviço (*Quality of Service* - QoS) [Sadok e Kamienski, 2000].

A engenharia de tráfego em redes MPLS é referenciada como MPLS-TE (MPLS – *Traffic Engineering*) [Systems, 2001]. O MPLS-TE permite a construção manual ou dinâmica de túneis que funcionam como caminhos da rede MPLS. O túnel é identificado por um rótulo por onde passam os fluxos de dados. As operadoras utilizam túneis para balanceamento de tráfego e para proteção de determinados fluxos com túneis primários (*primary tunnels*) e de proteção (*backup tunnels*). A Figura 5.1 mostra o balanceamento de tráfego utilizando MPLS-TE, bem como o desacoplamento das informações de enlaces físicos com a camada IP/MPLS. Esse desacoplamento é um problema atual, já que exige a configuração manual de túneis. Outro problema ocorre em caso de falhas nos enlaces físicos por longos períodos, já que torna-se necessário reconfigurar todos os túneis que passam pelo enlace defeituoso. Além disso, em geral, o plano de controle da rede IP/MPLS executa em hardware proprietário e de alto custo.

Os diferentes problemas fizeram com que, com o passar do tempo, o uso do MPLS nas redes de operadoras se tornasse um entrave. Nesse sentido, o paradigma das Redes Definidas por Software (*Software Defined Networks* - SDN) torna-se atrativo. Nas redes SDN, com a centralização do controle, é possível programar o roteamento da rede IP/MPLS por fluxo baseado nos objetivos da engenharia de tráfego. Nesse caso, a programação dos caminhos dos fluxos da rede pode ser realizada considerando, inclusive, mais de um parâmetro de configuração, como banda disponível, latência e grupos de risco compartilhado de enlaces físicos. Os roteadores de núcleo tornam-se comutadores "programáveis" e toda a lógica de programação é transferida para o controlador SDN, que pode executar em servidores de uso comum (*Commercial off the shelf* - COTS) e não mais

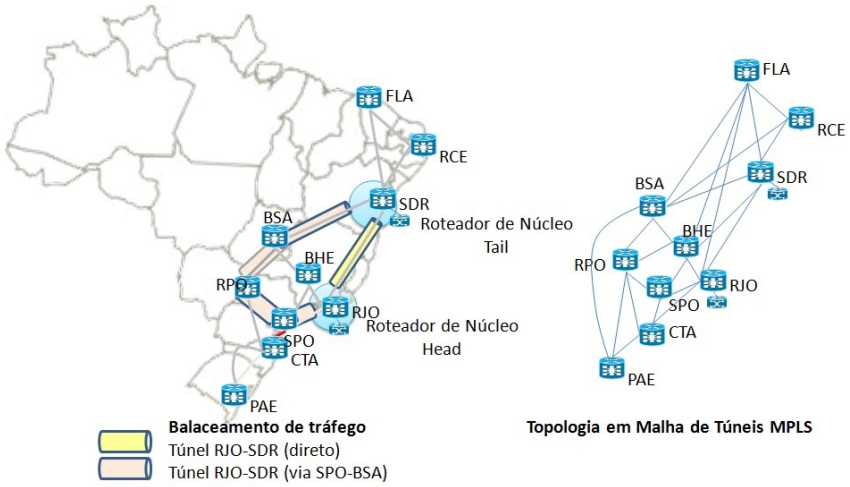


Figura 5.1. Configuração manual de túneis MPLS-TE.

em hardwares proprietários, de alto custo [Kreutz et al., 2015]. As interfaces (*northbound* e *southbound*) comunicam-se com o gerenciamento da rede através de APIs (*Application Programming Interfaces*) abertas ou proprietárias. Essa simplificação reduz as limitações da configuração de túneis TE das redes MPLS.

O roteamento por segmentos é uma proposta para simplificação do roteamento e configuração, mantendo estados por fluxo apenas nos nós de borda da rede, apoiando-se na programação das redes SDN. Em outras palavras, o roteamento por segmentos emprega roteamento pela fonte. Assim, o roteador de entrada da rede possui uma lista ordenada dos segmentos que definem o caminho na rede até o roteador de saída. O cálculo do caminho é feito pelo Elemento de Computação de Caminhos SDN (*SDN Path Computation Element*), ou através de uma aplicação SDN, podendo considerar objetivos de engenharia de tráfego. Em uma rede IP/MPLS, os segmentos são rótulos MPLS, que agrupados nos roteadores de núcleo de borda, informam o caminho do fluxo. Assim, apenas os roteadores de borda contêm a informação dos rótulos, não sendo necessário mantê-la nos roteadores intermediários. Essa característica permite reduzir a complexidade do plano de controle dos roteadores de núcleo.

O restante deste capítulo está organizado da seguinte forma. A Seção 2 introduz os problemas atuais das redes das operadoras na configuração e operação de redes IP/MPLS. A Seção 3 aborda o roteamento em redes SDN, apresentando uma visão geral da arquitetura SDN, detalhando as interfaces northbound e southbound usadas no roteamento por segmentos. A Seção 4 descreve os conceitos básicos do roteamento por segmentos, o funcionamento do plano de dados MPLS com roteamento por segmentos, e o plano de controle IGP para o roteamento por segmentos. Também são mostrados exemplos de roteamento por segmentos em IPv6 e a interoperabilidade de redes IP/MPLS legadas. Na Seção 5 são abordados simuladores de redes SDN, em especial o simulador Mininet e experimentos com controlador SDN OSHI SRTE (*Open Source Hybrid IP/SDN networking with Segment Routing and Traffic Engineering*). A Seção 6 conclui este minicurso destacando outras iniciativas e direções futuras.

5.2. Limitações das Tecnologias das Redes de Núcleo

As limitações da comutação de pacotes IP exigiram da indústria o desenvolvimento de tecnologias que trouxessem desempenho às redes de roteadores. As soluções inicialmente propostas utilizavam comutadores ATM com roteamento baseado em gerência de redes, evoluindo para o desenvolvimento de um hardware que ampliasse a capacidade de comutação, separando os protocolos de controle da comutação de pacotes, sendo esta última, o embrião das redes MPLS. A tecnologia MPLS [Rose, 2014] surgiu como uma proposta de evolução das redes públicas de comutação e encaminhamento de pacotes, motivada pela convergência da comunicação digital (voz, dados e vídeo) em uma infraestrutura comum. A ideia era integrar o MPLS com as redes IP e com outros tipos de redes legadas como o Ethernet, o Frame Relay e o próprio ATM [El-Sayed e Jaffe, 2002].

A rede IP/MPLS tem planos de controle e de encaminhamento de dados implementados em cada nó da rede. O plano de controle é mantido através de protocolos como o LDP (*Label Distribution Protocol*) para distribuição dos rótulos e um protocolo IGP, como o OSPF (*Open Shortest Path First*), para descoberta dos caminhos mais curtos denominados LSP (*Label Switched Path*). No caso de falha da rede, o OSPF encontra um novo caminho, desviando o tráfego para este [Santos et al., 2007, De Ghein, 2007].

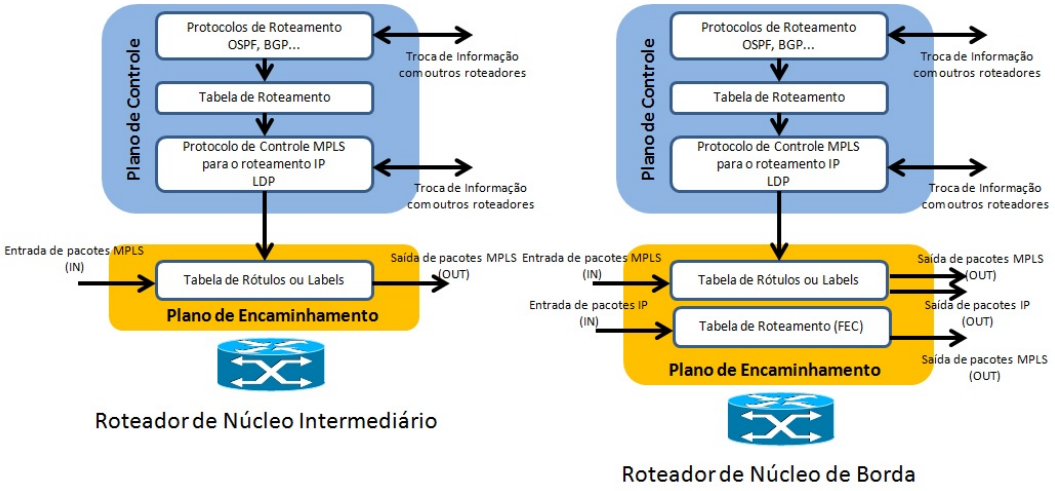
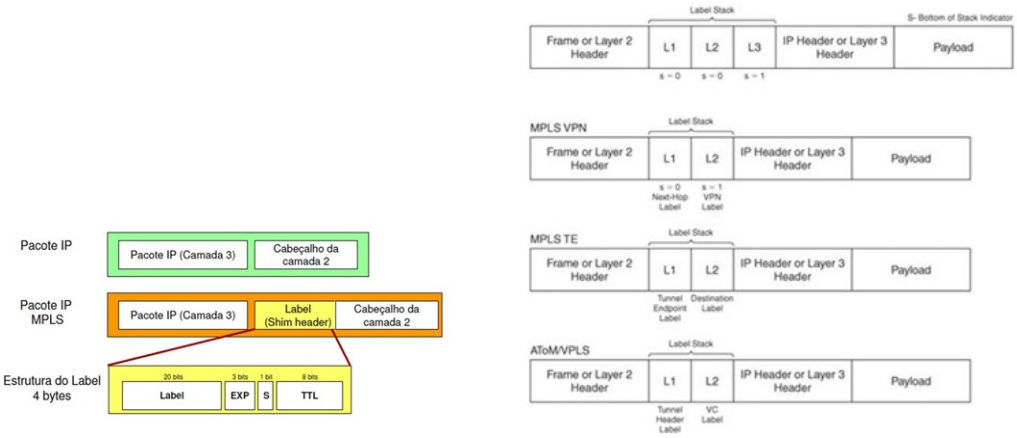


Figura 5.2. Representação de um roteador IP/MPLS.

5.2.1. Fundamentos do MPLS

Em uma rede MPLS, o pacote IP recebe um rótulo (*label*), inserido entre o cabeçalho IP e o cabeçalho do protocolo de camada inferior. Um rótulo MPLS possui 32 bits. Os primeiros 20 bits são o valor do rótulo. Os três próximos bits são experimentais, usados para marcação de classes de serviço. Esses bits são seguidos pelo bit S (*Bottom of Stack*), cujo valor é 0 se o rótulo é o inferior da pilha de rótulos ou 1 caso contrário. Os últimos 8 bits servem para limitar o tempo de vida do pacote (*Time to Live* - TTL) de forma similar ao IP [Marzo et al., 2003]. A Figura 5.3(a) ilustra o rótulo MPLS.

Os roteadores MPLS podem processar um conjunto de rótulos no pacote, organizados em pilha sendo o primeiro denominado de externo ou superior *(top label)*



(a) Estrutura do rótulo MPLS. (b) Empilhamento de rótulos MPLS.

Figura 5.3. Encapsulamento MPLS.

e o último rótulo, interno ou inferior *(bottom label)* (Figura 5.3(b)). Assim, são possíveis aplicações como redes privativas virtuais (*Vitual Private Networks* - VPN) e linhas alugadas virtuais (*Virtual Leased Lines* - VLL ou *Virtual Private LAN Service* - VPLS) [Sadok e Kamienski, 2000]. O transporte de outras tecnologias sobre o MPLS (*AToM - Any Transport Over MPLS*) é uma característica fundamental para o Roteamento por Segmentos [Pepelnjak e Guichard, 2003, Asati, 2012].

A tecnologia MPLS introduziu mecanismos como qualidade de serviço, engenharia de tráfego [Sadok e Kamienski, 2000], bem como novos serviços do tipo VPN na rede de núcleo das operadoras de telecomunicações. Existem dois tipos principais de VPN: de camada 2 (*VPN Layer 2*), por exemplo, o Ethernet ou o ATM, e o de camada 3 *(VPN Layer 3)*, que transporta protocolos de camada 3, mais especificamente, o IP. A rede IP/MPLS é muitas vezes considerada como uma rede multisserviço, já que oferece suporte a redes IP, VPN, Frame Relay, etc.

5.2.2. Arquitetura de Redes IP/MPLS

A arquitetura de uma rede IP/MPLS é composta por roteadores de borda denominados PE (*Provider Edge*) ou LER (*Label Edge Router*) que inserem e retiram rótulos dos pacotes. Os roteadores intermediários (P – *Provider* ou LSR – *Label Switch Router*) realizam a comutação de rótulos, enviando o pacote enlace a enlace. O CE (*Customer Edge*) representa o roteador IP que pertence a um determinado sistema autônomo [De Ghein, 2007, Systems, 2001]. A sequência de rótulos que forma o circuito virtual é um LSP (*Label Switched Path*), unidirecional (Figura 5.4). Ao separar o plano de controle do plano de encaminhamento de pacotes, o MPLS permite que novas facilidades sejam inseridas no plano de controle, sem necessidade de criar um novo plano de encaminhamento. Essa é a base para que aplicações como a engenharia de tráfego sejam criadas a partir do encaminhamento baseado em rótulos. A tabela FEC (*Fowarding Equivalence Class*) associa o endereço IP de destino com o rótulo de entrada na rede MPLS.

Os roteadores de borda (LER) recebem os pacotes IP provenientes do roteador

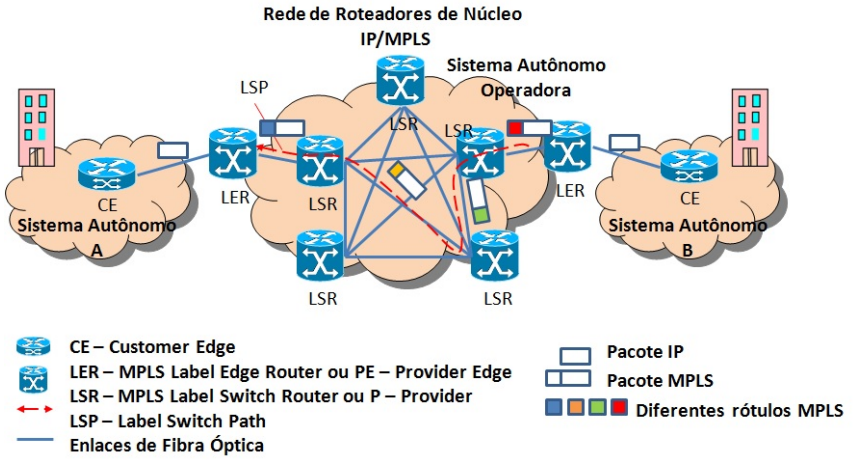


Figura 5.4. Arquitetura de Rede IP/MPLS.

cliente (CE) de origem, inserindo/retirando um rótulo adequado ao seu IP de destino e classe de serviço. Essas operações denominam-se inserção/remoção de rótulo (*push/pop*). No interior da rede, o pacote é comutado nos roteadores LSR intermediários apenas com base na informação dos rótulos. A informação de rótulos em uso está contida em uma base de instâncias de rótulos de encaminhamento (*Label Forwarding Information Base* - LFIB) em cada um dos roteadores intermediários do núcleo da rede. Na outra extremidade, o roteador MPLS de borda faz a operação inversa, retirando o rótulo e entregando o pacote para a rede do roteador cliente de destino. O processo de comutação de rótulos é mostrado na Figura 5.5. A tabela FEC associa o endereço IP de destino com o rótulo de entrada e saída na rede MPLS. O protocolo LDP (*Label Distribution Protocol*) é responsável pela distribuição de rótulos, trocando informações da FEC e rótulos associados entre os roteadores MPLS de borda e intermediários. Dessa forma, procura-se manter a coerência entre os LSPs formados e os prefixos IP dos pacotes encaminhados [Rose, 2014].

As tabelas FEC (Figura 5.5) são preenchidas por algum protocolo de roteamento (ex. OSPF, BGP) entre o CE de origem e o de destino. O protocolo LDP identifica seus vizinhos através de mensagens `HELLO`. Para cada entrada da tabela de roteamento é criada uma entrada na FEC, associando um rótulo de entrada. Os roteadores de núcleo (LERs e LSRs) anunciam aos seus vizinhos o par FEC/Rótulo de Entrada, sendo esta operação denominada distribuição de rótulos. Cada roteador de núcleo monta uma tabela completa com FEC/Rótulo de Entrada/Rótulo de Saída/Porta, denominada LFIB (Figura 5.5). Dessa forma, o plano de encaminhamento é responsável pela comutação dos pacotes baseado apenas nos rótulos envolvendo as operações de comutação, inserção e retirada de rótulos. No plano de controle, o protocolo LDP traduz as tabelas de rótulos em informações da tabela de roteamento. Ainda no plano de controle, o cálculo de caminhos na rede MPLS é feito por um IGP.

5.2.3. Serviços de Rede IP/MPLS

O principal serviço em redes IP/MPLS é o VPN, que pode ser configurado em topologias do tipo malha completa ou parcial, ou matriz-filial (*hub and spoke*). O MPLS

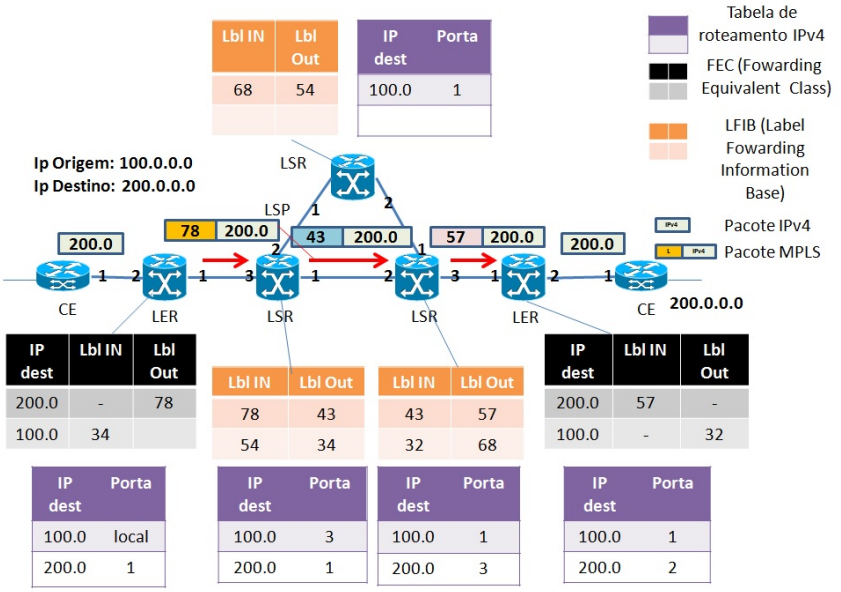


Figura 5.5. Comutação MPLS.

permite dois tipos de VPN [Pepelnjak e Guichard, 2003, Asati, 2012], a VPN de camada 3 (VPN L3) e a de camada 2 (VPN L2).

As VPNs L3 permitem a troca de protocolos de roteamento "*peering*" entre os roteadores cliente (CE) e os roteadores de borda do núcleo (LER). As portas do LER são associadas a uma VPN através de uma instância de roteamento chamada VRF (*Virtual Routing and Forwarding tables*). As VRFs são utilizadas para definir os LSPs. Como cada VPN utiliza sua própria VRF, os CEs de diferentes VPNs podem ter o mesmo endereço IP. Os CEs utilizam protocolos de roteamento tradicionais, como OSPF, RIP e BGP, com os LERs aos quais estão diretamente conectados. Os roteadores de borda (LER) trocam informações aprendidas pelos roteadores de cliente (CEs) diretamente conectados através do protocolo MP-BGP (*MultiProtocol-BGP*). Na percepção do roteador do cliente, a rede IP/MPLS funciona como uma rede IP dedicada. O processo de encaminhamento das VPNs utiliza dois rótulos, ou seja, o roteador de núcleo de borda insere dois rótulos MPLS entre o cabeçalho de camada 2 e o cabeçalho IP, empilhando rótulos. O rótulo mais externo refere-se ao roteador de núcleo (LER) de destino, enquanto o rótulo mais interno refere-se à VPN. O MPLS também possui a funcionalidade PHP (*Penultimate Hop Popping*) onde o penúltimo LSR (*penultimate LSR*) retira o rótulo mais externo antes de entregar o pacote ao LER. Esse processo é útil em VPNs L3, pois reduz a carga de processamento do LER, eliminando um rótulo para ser processado. O LER anuncia o rótulo de valor 3 que significa a funcionalidade de PHP. Esse rótulo é chamado *implicit-null*, que quando usado leva à perda das informações de QoS dos bits experimentais. A solução é o uso de rótulos especiais denominados *explicit-null* (valor 0), que é lido pelo LER para fins de QoS, mas removido da FIB. A Figura 5.6 ilustra uma VPN L3.

As VPNs L2 trocam com os roteadores de núcleo apenas informações de camada 2, sendo as informações de camada 3 trocadas apenas entre os CEs. As VPNs L2 podem

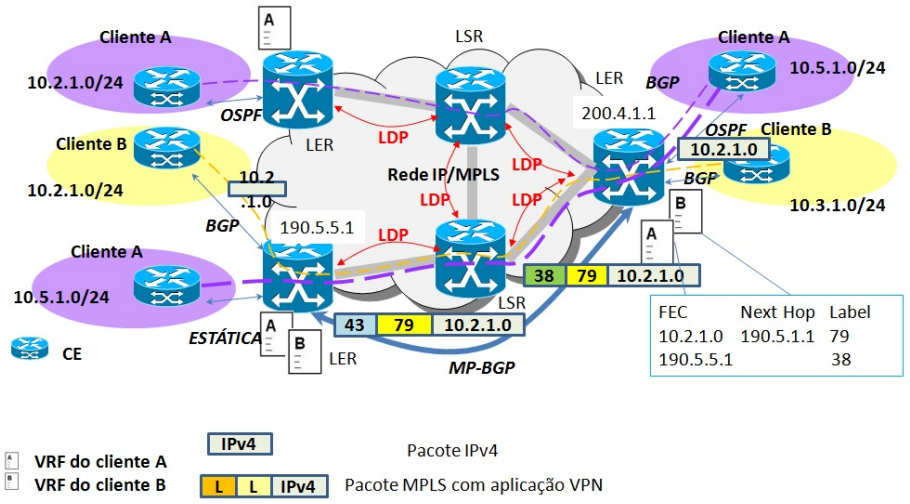


Figura 5.6. Serviço de rede privativa VPN L3.

ser entendidas como um enlace virtual, que pode ser ponto-a-ponto, no caso do VLL; e ponto-multiponto, no caso do VPLS. O transporte de quadros L2 introduziu o conceito de circuito virtual (*Virtual Circuit* - VC) no MPLS. Um LSP age como um túnel que permite carregar múltiplos VCs, enquanto o VC permite o transporte de quadros L2. O VC é implementando utilizando empilhamento de rótulos [IETF MPLS documents, 2001]. A Figura 5.7 mostra uma VPN L2 ponto-a-ponto, com um LSP conectando os roteadores de borda dos clientes A e B. O roteador de núcleo de borda A informa ao B que o roteador A receberá pela porta 1 os quadros L2 com outro rótulo, enviando os quadros com dois rótulos, fazendo com que chegue ao seu destino na porta correta.

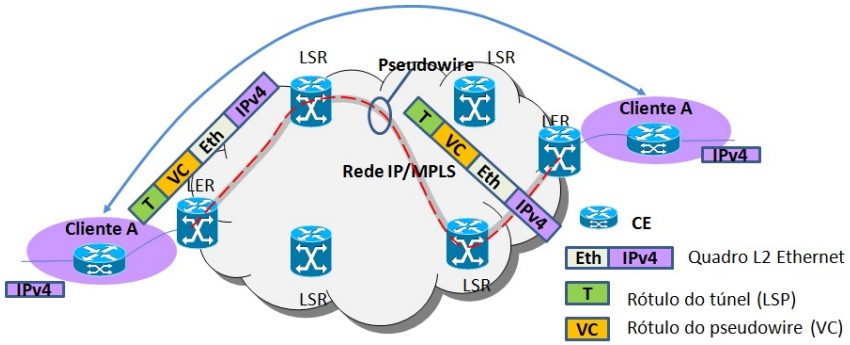


Figura 5.7. Serviço de rede privativa VPN L2 ponto a ponto (VLL).

5.2.4. QoS, Engenharia de Tráfego e Restauração em Redes IP/MPLS

O protocolo MPLS, assim como o protocolo IP, permite utilizar o modelo de Diff-Serv (*Differentiated Services*) [Marzo et al., 2003] para implementação de QoS. O Diff-Serv gerencia os recursos de rede e cria classes de serviço atendidas em filas diferentes conforme o nível de prioridade do fluxo de dados [IETF MPLS documents, 2001]. Enquanto no IPv4 a classe de serviço é indicada no campo DSCP (*Differentiated Services*

Code Point, anteriormente, ToS – *Type of Service*) [Marzo et al., 2003], no MPLS, a informação é definida no campo EXP bits do rótulo.

O conceito de engenharia de tráfego, MPLS-TE (*MPLS – Traffic Engineering*), foi desenvolvido para resolver problemas de congestionamento, otimização da banda e convergência da rede com proteção do tráfego por reserva de recursos. A estratégia do MPLS-TE é calcular caminhos baseados em restrições como banda disponível, latência e perda de pacotes, sugerindo uma rota diferente do IGP, que tipicamente não realiza engenharia de tráfego [Xiao et al., 2000, Alvarez, 2016]. O cálculo de caminhos baseados em restrições é feito pelo PCE (*Path Computation Element*), papel desempenhado pelo plano de controle dos roteadores de núcleo (Figura 5.8). O caminho calculado pela engenharia de tráfego é implementado como um túnel unidirecional (túnel TE) iniciado no roteador de núcleo de origem (*head end LSR*) e terminado no roteador de núcleo de destino (*tail end LSR*). O túnel é identificado a partir de um rótulo a mais além dos rótulos do protocolo LDP. A reserva de recursos para o caminho é feita pelo protocolo RSVP-TE (*Resource reSerVation Protocol – Traffic Engineering*), uma extensão do RSVP [Rose, 2014]. O RSVP-TE é usado para o estabelecimento de LSPs, que podem ser configurados manualmente (*Explicit LSP*) ou dinamicamente (*Dynamic LSP*) através de um IGP com extensões para engenharia de tráfego, como o OSPF-TE [Rose, 2014]. Na Figura 5.8, o roteador R1 utiliza a base de dados da topologia TE, que contém múltiplas métricas por enlace, para calcular o caminho até R8 através do PCE presente em R1. O RSVP-TE cria LSPs adicionais distribuindo os rótulos entre os roteadores R1 e R8. As mensagens RSVP-TE de caminho são enviadas periodicamente pelo roteador de núcleo de origem para cada roteador de núcleo intermediário pertencente ao túnel, contendo uma lista de atributos a serem analisados. Esses atributos possuem os valores das métricas utilizadas no cálculo dos caminhos. As mensagens de reserva de recursos são enviadas pelo roteador de destino após receber uma mensagem de estabelecimento de caminho, iniciando o processo de distribuição de rótulos, no qual cada roteador intermediário informa o rótulo a ser usado pelo antecessor. Se um dos roteadores intermediários não puder confirmar a reserva, uma mensagem de erro é enviada ao roteador de núcleo de origem do túnel [Alvarez, 2016].

Um dos requisitos da engenharia de tráfego é a capacidade de re-rotear um túnel TE baseado em políticas administrativas. O mecanismo de re-roteamento rápido (*Fast Reroute* – FRR) previne falhas de nós e de enlaces. O mecanismo pode ser usado para trocar para rotas com melhor desempenho, em caso de falhas de recursos do túnel, ou utilizar novas rotas por decisões administrativas. A Figura 5.9 mostra a criação do túnel primário formado pelos roteadores A, B, D, E e do túnel de proteção, provisionado nos roteadores B, C e D [Osborne e Simha, 2002, Alvarez, 2016]. A engenharia de tráfego também pode ser usada para balanceamento de tráfego através de múltiplos túneis TE. A malha de túneis e sua banda podem ser configurados manual ou automaticamente, através de um roteiro repetitivo de configurações [Osborne e Simha, 2002]. Em uma topologia em malha, tanto para balanceamento de tráfego, quanto para proteção, os caminhos precisam ser disjuntos. Entretanto, alguns túneis podem utilizar o mesmo recurso físico, como o mesmo cabo de fibra óptica. No exemplo da Figura 5.10, o SRLG (*Shared Risk Link Group*) de número 10 indica a presença de um enlace físico comum entre os caminhos R2-R4/R2-R3. A informação do SRLG é usada como um atributo adicional ao caminho original fornecido pelo IGP para evitar a configuração de túneis no mesmo enlace físico [Alvarez, 2016].

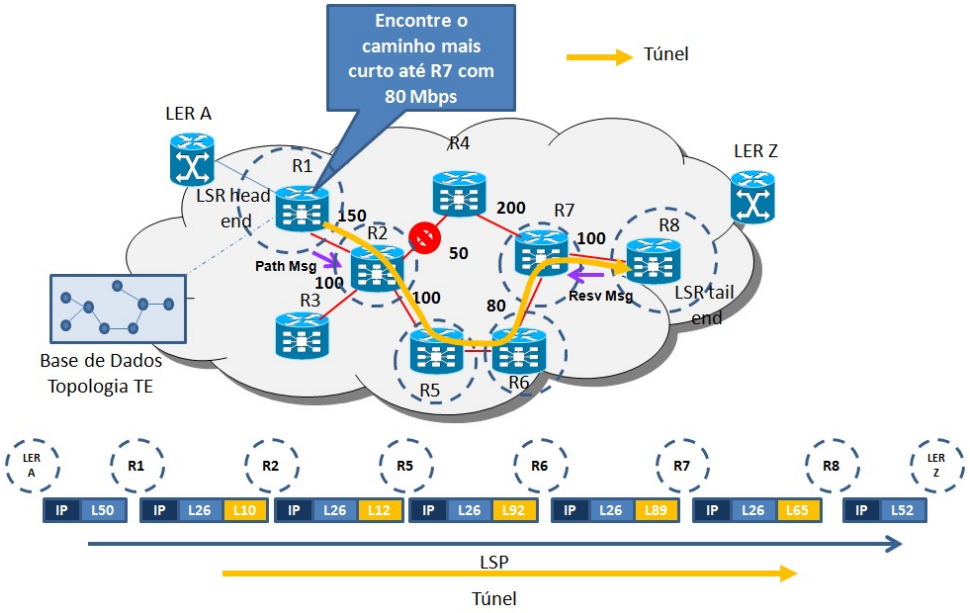


Figura 5.8. Cálculo de caminho do túnel TE.

5.2.5. Desafios para Engenharia de Tráfego e Restauração em Redes IP/MPLS

A operadora de telecomunicações utiliza a engenharia de tráfego para o tratamento de falhas nos enlaces físicos da rede. Tipicamente, a camada física ou rede de transporte é composta por equipamentos DWDM (*Dense Wavelength Division Multiplex*), ROADM (*Reconfigurable Add Drop Multiplex*) e comutadores OTN (*Optical Transport Network*). Esses elementos realizam a multiplexação e comutação do tráfego no domínio óptico e elétrico, além de realizar funções de formatação e regeneração dos sinais, de proteção e de restauração da rede. A rede de transporte possui um plano de controle próprio baseado no padrão ASON (*Automatically Switched Optical Network*) [Je e Ly, 2012] e GMPLS (*Generalized MPLS*) [J., 2011].

A rede de transporte é segregada da rede de roteadores de núcleo (camada L3), sendo necessária a configuração de SRLGs para evitar túneis primários e de proteção em um mesmo enlace físico. É comum o cenário com proteções e restaurações em ambas as redes, ocasionando desperdício da banda disponível. A Figura 5.11 ilustra a separação entre as redes de transporte e de roteadores de núcleo (rede IP/MPLS). Essa separação evidencia o desacoplamento dos planos de controle de cada camada, resultando em uma ocupação não otimizada da banda disponível como resultado da configuração de proteção de múltiplas camadas.

A configuração manual ocorre frequentemente devido às mudanças da matriz de tráfego e da topologia física, muitas vezes como decorrência de falhas nos enlaces. Em qualquer dos casos, a malha de túneis deve ser reconfigurada para se adequar ao novo padrão de tráfego. Outro problema das operadoras é o tempo de convergência da rede em caso de falhas, intrinsecamente relacionado à separação da rede de roteadores de núcleo (camada L3) da rede de transporte (camadas L0, L1 e L2). A desvinculação da topologia

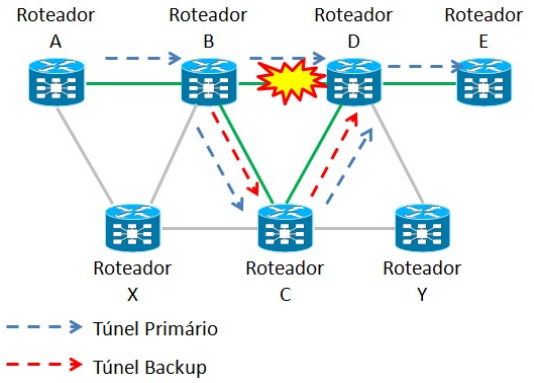


Figura 5.9. Túnel primário e de proteção.

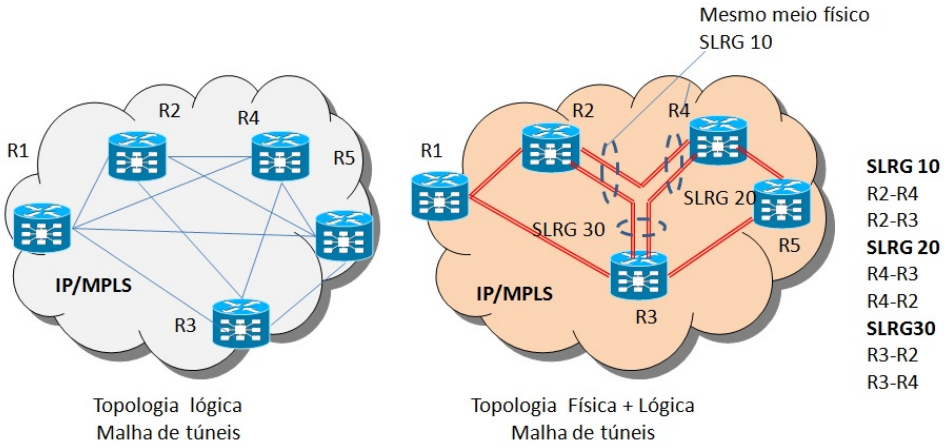


Figura 5.10. SLRG (*Shared Risk Link Group*).

de rede física da rede lógica de túneis TE é um desafio enfrentado pelas operadoras, pois uma interrupção em um enlace físico pode corresponder à interrupção de ambos os túneis, o TE principal e reserva.

5.3. Conceitos de Redes Definidas por Software

O Roteamento por segmentos utiliza aplicações de Redes Definidas por Software (SDN) para o cálculo dos caminhos dos túneis MPLS, baseado nos parâmetros de engenharia de tráfego. A configuração do túnel é representada por uma lista ordenada de segmentos. As SDNs constituem uma mudança de paradigma em redes: separando o plano de controle do plano de encaminhamento, rompe a integração vertical comum em roteadores e comutadores. A centralização do controle da rede por outro lado introduz maiores capacidade de programação e flexibilidade de utilização [Kreutz et al., 2015, Hu et al., 2014].

5.3.1. Arquitetura e Desenvolvimento da SDN

Em redes IP tradicionais os plano de controle e de dados são acoplados e embutidos no mesmo dispositivo de rede. Essa integração vertical é uma das razões pelas quais

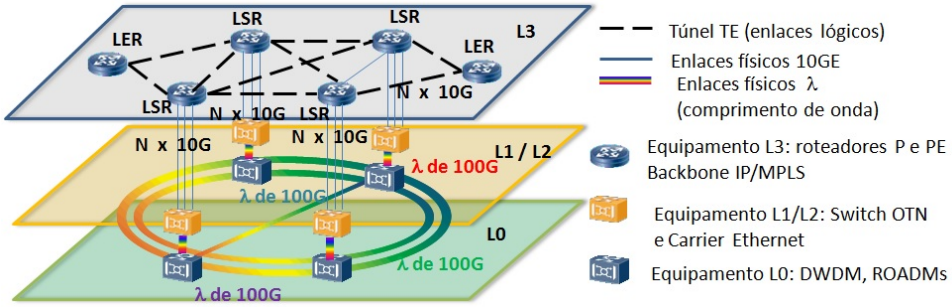


Figura 5.11. Separação de camadas de rede de transporte e IP/MPLS.

as redes IP possuem gerenciamento e controle complexos, sendo ainda pouco flexíveis à introdução de inovações tecnológicas. Para gerência de redes, a maior parte dos fabricantes oferece soluções com hardware, sistemas operacionais e programas de controle proprietários. As redes de núcleo IP/MPLS apresentam também grande complexidade de configuração devido aos inúmeros protocolos envolvidos, bem como a falta de flexibilidade em resposta a mudanças da rede, falhas e balanceamento de tráfego.

A arquitetura de rede SDN se baseia em quatro pilares: desacoplamento do plano de controle do plano de dados; decisões baseadas em fluxos ao invés do endereçamento de destino; programação de fluxos flexível, limitada ao tamanho das tabelas de fluxos; e lógica de controle em entidade externa denominada controlador. O controlador é uma plataforma de software ou um sistema operacional de rede (*Network Operating System* - NOS) executado em um servidor de uso comum COTS. O objetivo é prover os recursos essenciais de rede e abstrações para facilitar a programação de qualquer dispositivo de encaminhamento. Em uma visão simplificada da arquitetura SDN, a infraestrutura física é formada pelos dispositivos de encaminhamento, como comutadores ou roteadores sem plano de controle ou com o plano de controle reduzido e com interfaces de programação abertas. A Figura 5.12 mostra a arquitetura de uma SDN [Kreutz et al., 2015]. O plano de dados é constituído pela infraestrutura de rede e interfaces *southbound* (SBI – *southbound interface*). A infraestrutura de rede, similar a uma rede tradicional, corresponde aos dispositivos que desempenham instruções elementares de encaminhamento de tráfego. Estas instruções são definidas por interfaces abertas SBI, como por exemplo o OpenFlow [Lara et al., 2014]. O plano de controle é constituído por uma camada de virtualização, outra camada correspondente ao sistema operacional de rede e as interfaces *northbound* (NBI). O sistema operacional de rede pode estar ou não em uma infraestrutura de hardware virtualizada através do hipervisor, que permite a criação de máquinas virtuais no servidor físico onde é instalado o controlador de rede. O controlador ou sistema operacional de rede programa os dispositivos de rede. Por fim, as interfaces *northbound* oferecem APIs para desenvolvedores de aplicações. O plano de gerência corresponde ao conjunto de aplicações, utilizando linguagens de programação apropriadas para o ambiente virtualizado, que interliga todas as funcionalidades de rede para que as aplicações possam implementar a lógica e controle da rede através da NBI. O plano de dados é responsável pelo encaminhamento dos pacotes, e é composto pela infraestrutura de rede, que corresponde ao elementos físicos da rede e por interfaces NBI responsáveis pela comuni-

cação entre o plano de controle e o plano de dados. A seguir serão detalhadas cada uma das camadas da arquitetura SDN.

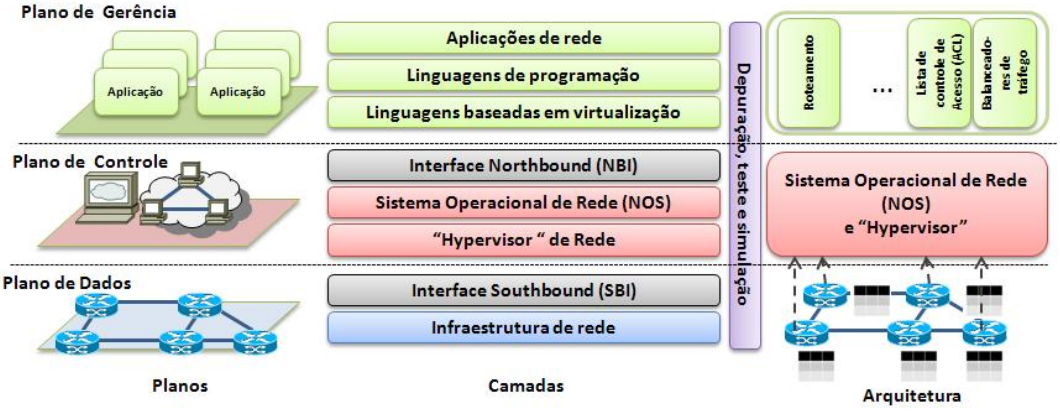


Figura 5.12. Diagrama de arquitetura de sistema de rede SDN.

5.3.2. Plano de Dados

O plano de encaminhamento de dados é composto pela infraestrutura de rede física e pelas interfaces *southbound* onde protocolos abertos ou proprietários podem ser usados para programação dos dispositivos de rede, como o CWMP (*CPE WAN Management Protocol*) definido na norma ETSI TR-069 ou o protocolo SNMP (*Simple Network Management Protocol*) padronizado pelo IETF. O SNMP foi concebido para gerenciamento de dispositivos de rede através de operações do tipo "GET" e "SET" de informações em uma base de dados (*Management Information Base* – MIB).

5.3.2.1. Infraestrutura de Rede

A infraestrutura física das redes SDN é similar à das redes tradicionais, com a diferença dos dispositivos serem simplesmente elementos de encaminhamento de dados [Nunes et al., 2014], ou seja, sem controle ou software embarcado que permita o dispositivo tomar decisões autônomas. A inteligência reside no sistema operacional de rede e nas aplicações de rede, que acessam os elementos da infraestrutura através das interfaces *southbound*, como por exemplo o OpenFlow.

5.3.2.2. Interfaces Southbound: NETCONF, PCEP, BGP-LS, OpenFlow e outros

Atualmente, muitas interfaces abertas podem ser empregadas como interface *southbound* em redes SDN. Além do OpenFlow, pode-se utilizar os protocolos NETCONF [RFC6241, 2011], BGP-LS (*Border Gateway Protocol - Link State*) [RFC7752, 2016a], PCEP (*Path Computation Element Communication Protocol*) [RFC5440, 2009], OVSDB (*Open vSwitch Database*) [RFC7047, 2013], SNMP (*Simple Network Management Protocol*) [RFC1157, 1990], a TL1 (*Transaction Language 1*) [Telcordia GR-831, 1996] e a CLI (*Command Line Interface*) [ISO/IEC 23271:2012, 2012]. A linguagem TL1 foi

desenvolvida especificamente para equipamentos de telecomunicações, cujo objetivo é a comunicação entre máquinas. O protocolo NETCONF foi desenvolvido para ser o sucessor natural do SNMP, pois o SNMP tinha o foco no monitoramento e não na configuração da rede. O OVSDB é um protocolo de gerenciamento projetado para redes definidas por software. Originalmente o OVSDB era parte do OVS (*Open vSwitch*), um comutador virtual projetado para hipervisores Linux. O OVS representa uma evolução dos protocolos de gerenciamento de rede, permitindo programar e configurar pontes, portas e interfaces de plataformas de equipamentos SDN e de virtualização de funções de rede (*Network Functions Virtualization* – NFV) [Andreas et al., 2015]. A seguir são detalhadas os principais protocolos de interfaces SBI.

NETCONF e YANG

O NETCONF é um protocolo de gerenciamento de rede definido pelo IETF para configuração e monitoramento da rede, sendo este último papel desempenhado pelo SNMP desde o final dos anos 80. O SNMP tinha como ponto fraco a ausência de recursos de configuração de rede além da interface binária BER (*Basic Encoding Rules*) e MIBs proprietárias. O protocolo NETCONF utiliza mecanismos que permitem instalar, manipular e apagar a configuração de dispositivos de rede através de uma implementação cliente-servidor, ilustrada na Figura 5.13.

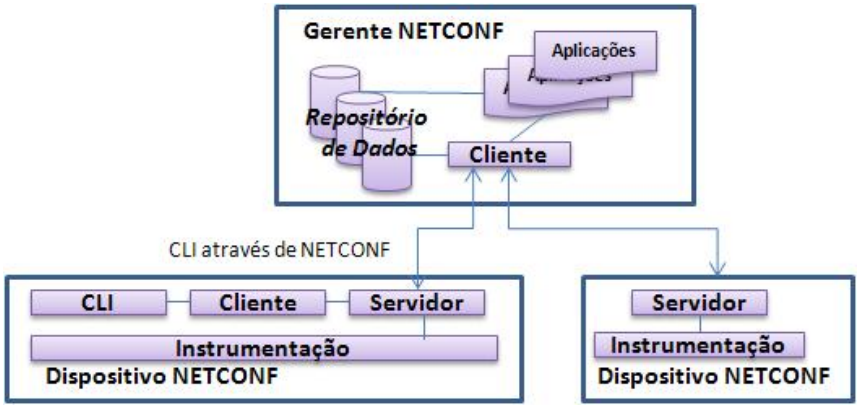


Figura 5.13. Modelo de implementação cliente-servidor do NETCONF.

Após o estabelecimento da sessão segura de transporte entre cliente e servidor, o protocolo NETCONF envia uma mensagem `HELLO` para anúncio das capacidades do protocolo e modelos de dados suportados. O NETCONF suporta ainda a subscrição e o recebimento de notificações de eventos de forma assíncrona assim como o fechamento parcial de uma configuração corrente de um dispositivo de rede. Esta funcionalidade permite múltiplas sessões de edição, agilizando o processo de configuração. O NETCONF permite o monitoramento e gerenciamento por uma entidade autônoma (o gerente NETCONF) que utiliza repositório de dados, sessões, fechamentos e estatísticas disponíveis no servidor NETCONF.

O YANG é uma linguagem formal com texto claro de modelo de dados com sin-

taxe e semântica que permitem a construção de aplicações de rede. O modelo YANG pode se traduzir em um arquivo de formato XML (*eXtensible Markup Language*) ou JSON (*JavaScript Object Notation*), estruturado em uma árvore para cada módulo, com propriedades que correspondem às funcionalidades do dispositivo; e declarações de tipos, dados, restrições e acréscimos de estruturas reutilizáveis [Nwa et al., 2010]. A Figura 5.14 mostra um diagrama representativo do modelo YANG, onde cada dispositivo de rede tem seus atributos descritos em um modelo de abstração, ilustrado na Figura 5.15. O protocolo NETCONF transporta estas informações até uma gerência de aplicações, e as aplicações podem inferir as configurações necessárias nos dispositivos de rede. A Figura 5.15 exemplifica um módulo YANG de inventário de interfaces um comutador OpenFlow. A estrutura em árvore do módulo representa os atributos de endereçamento da interface, com tipos de dados definidos [RFC6020, 2010].

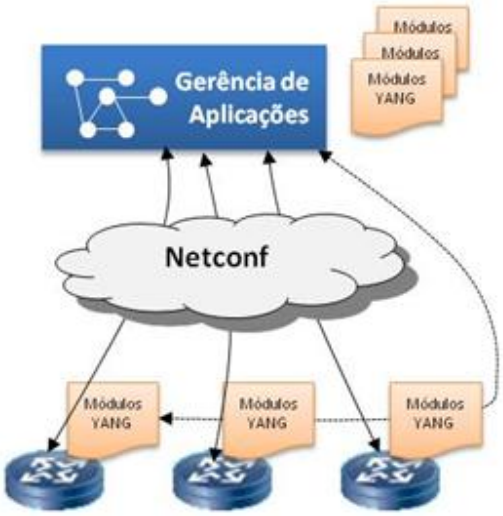


Figura 5.14. Modelo de dados YANG.

```
+--rw if:interfaces
 +--rw if:interface [name]
    ...
    +--rw ipv4
       +--rw enabled?            boolean
       +--rw ip-forwarding?      boolean
       +--rw address [ip]
          +--rw ip               inet:ipv4-address
          +--rw (subnet)?
             +--:(prefix-length)
             |  +--rw ip:prefix-length?   uint8
             +--:(netmask)
                +--rw ip:netmask?        inet:ipv4-address
```

Figura 5.15. Exemplo de módulo YANG para inventário de rede.

BGP-LS

Existem duas formas básicas de obtenção de informações de topologia de rede: protocolos de gerenciamento e roteamento. Um protocolo de roteamento responsável por obter a informação da topologia da rede é o BGP-LS (*Border Gateway Protocol - Link*

State), uma extensão do BGP [RFC7752, 2016b] que permite carregar informações dos estados dos enlaces. Essa informação é usada pelo IGP, que normalmente utiliza outra base de informações de estados dos enlaces como a TED (*Traffic Engineering Database*) usada na engenharia de tráfego, sendo que ambas provêem o mesmo conjunto de informações. No caso do BGP-LS, as informações podem ser agregadas de múltiplas áreas e de diferentes sistemas autônomos, permitindo uma análise abrangente do estado de toda rede. O BGP-LS foi desenvolvido especificamente para melhorar a escalabilidade do BGP como o controle baseado em fluxos TCP e no uso estratégico de roteadores RR (*Router Reflectors*). Em ambos os casos é necessário adquirir informações de topologia multiárea, o que é feito tradicionalmente por um elemento da rede de um sistema autônomo que reúne as informações dos demais elementos de outros sistemas autônomos através de meios manuais. Um controlador de engenharia de tráfego, por exemplo, uma aplicação de servidor PCE (*Path Computation Server* - PCS) implementa o BGP-LS para adquirir a topologia de rede a ser utilizada no roteamento da mesma [Casellas et al., 2015]. O BGP-LS suporta também mecanismos de políticas que limitam o uso de certos nós e enlaces da rede, ou seções de topologia particionadas pelo operador da rede. A Figura 5.16 mostra um controlador SDN interagindo com dispositivos de rede de diferentes ASes, com um IGP tradicional e com o BGP-LS. Note como o BGP-LS tem visão mutidomínio através do BGP-LS RR (*BGP-LS Router Reflector*) quando comparado à solução com IGP tradicional.

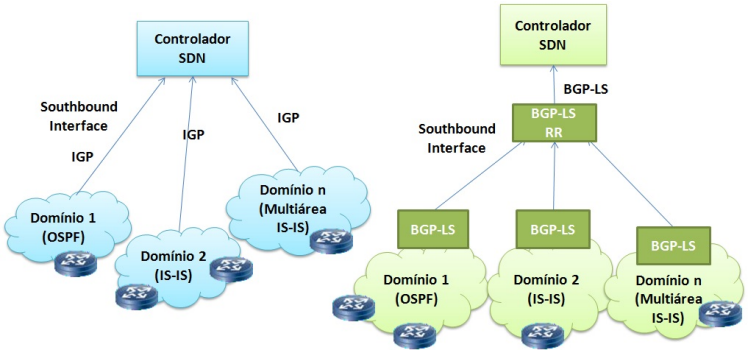


Figura 5.16. Southbound interface com IGP e com BGP-LS.

PCEP

O protocolo PCEP (*Path Computation Element Protocol*) é usado para comunicação entre o PCC (*Path Computation Client*) e o PCE (*Path Computation Element*), utilizando informações da base de estados do enlace, conforme a Figura 5.17. O PCE de controle dinâmico (*stateful PCE*) e o PCE iniciado pelo LSP são extensões ainda em discussão no IETF para habilitar o emprego destes protocolos em SDNs [Paolucci et al., 2013]. As extensões ainda não avançaram como uma proposição de padrão final devido à dificuldade de garantir nos padrões a orquestração entre PCEs de diferentes fornecedores [Casellas et al., 2015].

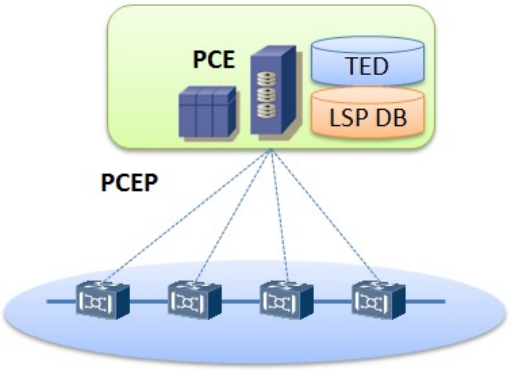


Figura 5.17. Arquitetura de Rede com Protocolo PCEP.

OpenFlow

Uma das principais tendências de SBI aberta para SDNs é o protocolo OpenFlow. O OpenFlow utiliza uma tabela com regras de tratamento de pacotes, na qual cada regra permite ações como encaminhamento, descarte e modificação do fluxo. O OpenFlow permite controlar os fluxos de dados encaminhando e processando os pacotes, conforme ações e regras configuradas pelo controlador nos comutadores OpenFlow. Os comutadores OpenFlow e o controlador da rede são interligados através de um canal de comunicação TLS (*Transport Layer Security*), conforme a arquitetura mostrada na Figura 5.18.

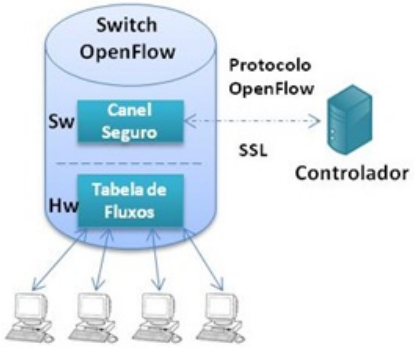


Figura 5.18. Arquitetura de uma rede OpenFlow.

A Figura 5.19 resume o funcionamento da SBI OpenFlow, com suas entradas na tabela de fluxo, cada uma definindo uma regra, uma ação e contadores de estatísticas. As regras são baseadas em informações do cabeçalho das camadas de enlace, rede e transporte. A partir da versão 1.1, o OpenFlow passou a suportar VLAN e Q-in-Q, portas virtuais e túneis, roteamento multi-percurso, ECMP (*Equal-Cost Multi-Path*) e MPLS.

5.3.3. Plano de Controle

O controlador representa o plano de controle em uma SDN [Xiea et al., 2015]. O controlador e as aplicações de rede podem ser instalados ou não em uma infraestrutura virtualizada. O controle da rede e as funções são abstraídas com a introdução de um

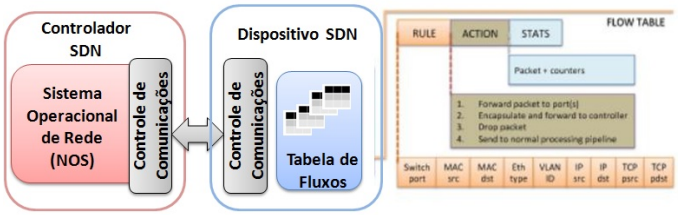


Figura 5.19. Tabelas de fluxos OpenFlow.

hipervisor [Andreas et al., 2015], criando uma rede independente do tipo de aplicação. O gerenciamento e orquestração da rede podem ser realizados por um orquestrador, como por exemplo, o OpenStack usado em nuvens computacionais [Huang et al., 2014]. A ideia é permitir que servidores e a infraestrutura da rede sejam virtualizados e implementados na nuvem [Matias et al., 2015]. O hipervisor permite a criação, remoção e movimentação das máquinas virtuais dos controladores SDN e de aplicações de rede. A Figura 5.20 mostra uma arquitetura comparativa de uma rede tradicional e de rede de SDN através de APIs existentes, onde o ambiente de virtualização é utilizado na camada de controle e gerenciamento, e finalmente a rede SDN com virtualização de funções de rede (NFV), denominado SDN overlay onde o SDN e a NFV atuam em conjunto [Xia et al., 2015].

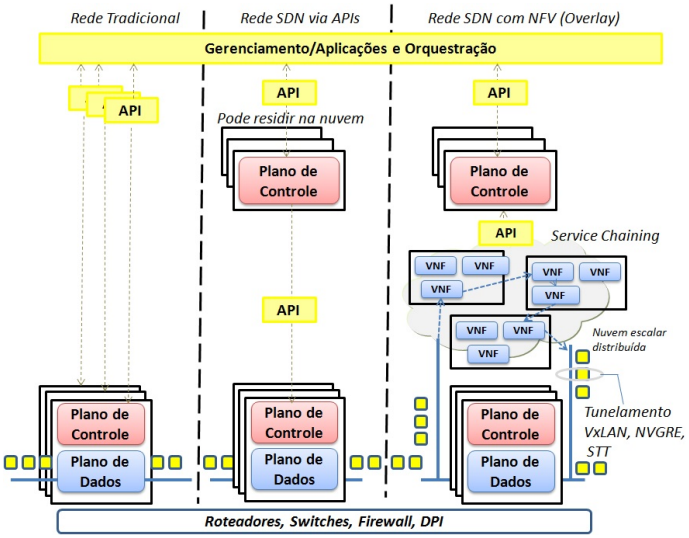
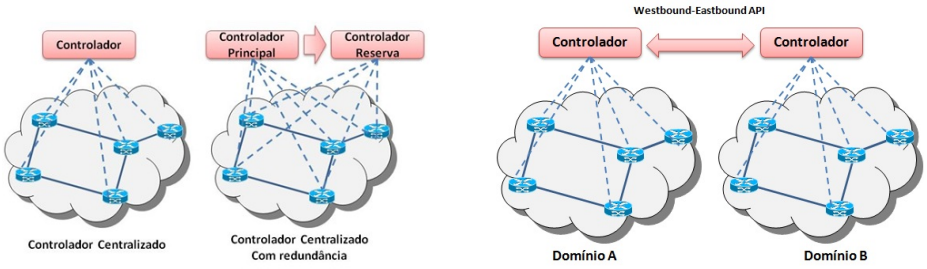


Figura 5.20. SDN via API e SDN Overlay.

5.3.3.1. Controladores SDN: Opendaylight, ONOS e outros

O controlador SDN, corresponde ao sistema operacional de rede, provêem APIs de alto nível para acessar um nível de abstração mais baixo dos dispositivos de rede. Tais instruções de nível mais baixo são específicas do dispositivo de rede e, na maioria das vezes, sistemas operacionais proprietários, como o IOS da Cisco ou o Junos da Juniper.

Atualmente, os projetistas de protocolos de roteamento precisam lidar com algoritmos distribuídos complexos para a solução de problemas em redes. As redes SDN facilitam o gerenciamento das redes e as soluções de problemas através da lógica centralizada. Assim, o controlador é o elemento crítico de uma arquitetura SDN. Existe uma grande diversidade de controladores e plataformas de controle com diferentes projetos e arquiteturas. As arquiteturas relevantes são com controladores centralizados ou distribuídos (Figura 5.21(a)). Um controlador centralizado tem a desvantagem de ser um ponto único de falhas, com escalabilidade limitada. No entanto, a centralização significa simplicidade de operação e melhor visão do comportamento da rede [Scott-Hayward, 2015]. Os controladores de código aberto NOX [Gude et al., 2016], POX [Kaur et al., 2016], Floodlight [Haleplidis et al., 2015] e Ryu [development team, 2016] são do tipo centralizado.



(a) Controlador centralizado e centralizado com redundância.

(b) Controlador distribuído..

Figura 5.21. Arquitetura de controladores.

Os controladores distribuídos podem ser organizados em *cluster* ou em um conjunto de controladores distribuídos em diferentes domínios de rede [Civanlar et al., 2015]. São exemplos de controladores distribuídos o Opendaylight [Medved et al., 2014] e o ONOS [Stancu et al., 2015]. Para controladores distribuídos em diferentes domínios, as APIs de fronteira leste e oeste (*eastbound-westbound*) [Pingping et al., 2015] são importantes. Atualmente, cada controlador implementa sua API de fronteira leste-oeste com o objetivo de troca de dados, verificação da consistência do modelo de dados e monitoramento de notificações. A maioria dos controladores distribuídos oferece uma consistência semântica baixa, ou seja, as atualizações de nós distintos são eventualmente atualizadas em todos os controladores. Isso implica períodos de tempo onde controladores distintos possuem visões diferentes para a mesma propriedade. A consistência forte, por outro lado, assegura que todos os controladores possuem a visão mais atualizada possível, depois de uma operação de escrita. Embora tenha impacto no desempenho do sistema, a consistência forte é uma aplicação de simples implementação. O controlador ONOS é um exemplo de controlador com consistência forte [Kreutz et al., 2015]. Atualmente existe um grande número de implementações de controladores SDN disponíveis, incluindo de código aberto ou comerciais. Os de código aberto possuem interfaces *northbound* (NBI) e *southbound* (SBI) abertas, permitindo a pesquisa de métodos inovadores de operação da rede [Rowshanrad et al., 2014]. Adicionalmente à pesquisa e experimentação, as interfaces abertas permitem que equipamentos de fabricantes diferentes possam interoperar. A seguir são enumerados alguns dos principais controladores existentes:

- **VMware/Nicira**: Plataforma de virtualização (*Network Virtualization Plataform* – NVP), atualmente comercializado como VMware NSX, utiliza um comutador virtual aberto (*Open Virtual Switch* – OVS) e o OpenFlow como interface SBI.

- **NOX/POX**: Controlador de código aberto que utiliza o OpenFlow 1.0. A pesquisa continuada pelo ON.LAB (*Open Networking Lab*) deu origem ao controlador ONOS aplicado atualmente na indústria de equipamentos de redes de telecomunicações. O controlador NOX foi desenvolvido em C++ não tendo sido implementado de forma massiva. O controlador POX foi o sucessor do NOX com interface gráfica escrita na linguagem Python, facilitando o desenvolvimento e experimentação.

- **Ryu**: Controlador com implementação em Python. Possui um serviço de mensagens de componentes implementadas em outras linguagens de programação, como por exemplo, bibliotecas do OpenFlow, gerenciamento de aplicações, serviços de infraestrutura e bibliotecas reutilizáveis como do protocolo NETCONF.

- **Beacon**: Controlador implementado em JAVA e integrado à IDE do Eclipse. O Beacon foi o primeiro controlador a criar um ambiente de trabalho de SDN, mas é limitado à topologia em estrela.

- **Big Switch Networks/Floodlight**: é uma ramificação do controlador Beacon. Foi inicialmente implementado usando o Apache Ant, uma ferramenta de desenvolvimento popular, tornando-o de fácil uso e flexível. O controlador Floodlight possui uma comunicadade ativa e um grande número de funcionalidades que podem ser adicionadas para requisitos específicos de uma empresa. Possui uma interface gráfica baseada em JAVA, e a maioria de suas funcionalidades está exposta em APIs REST.

- **Opendaylight**: é um projeto colaborativo da Linux Foundation, suportado pela Cisco e diversas outras empresas. Tal como o Floodlight, o Opendaylight foi escrito em JAVA. É orientado a API REST e interface web. Sua segunda versão (Helium) inclui suporte a NFV e redes de grandes dimensões. Também possui módulos plugáveis que podem ser utilizados de acordo com a necessidade. O Opendaylight é um pouco diferente dos demais controladores por oferecer outros protocolos como interfaces *southbound* além do OpenFlow, como o BGP e PCEP. Adicionalmente, o Opendaylight oferece interfaces com o Openstack e Open vSwitch (OVSDB). O Opendaylight é baseado em uma arquitetura de microsserviços, através do compartilhamento de estruturas de dados baseados em YANG para armazenamento de dados e troca de mensagens. Através de um modelo dirigido à camada de abstração de serviços (*Model Driven Service Abstraction Layer* - MD-SAL) qualquer aplicação ou função pode ser agregada a um serviço e carregada pelo controlador [Haleplidis et al., 2015].

- **ONOS**: controlador SDN voltado para operadoras de telecomunicações, projetado para alta disponibilidade, desempenho e escalabilidade através de instâncias distribuídas que conferem redundância ao controlador em caso de falha.

- **Opencontrail**: Controlador comercial da Juniper, baseado no Apache 2.0, com foco em NFV e com API REST.

Existem ainda outros controladores comerciais. Alguns exemplos são o Brocade Vyatta, o Cisco WAE (*WAN Automation Engine*), ambos baseados em Opendaylight com implementações próprias. O HP *Virtual Application Networks* (VAN) trabalha em conjunto com o Openstack com melhorias no controle do Open vSwitch e suporte a diferentes tipos de hipervisores e roteadores distribuídos [Kreutz et al., 2015].

5.3.3.2. Elemento de Computação – PCE

O PCE (*Path Computation Element*) é uma entidade que calcula caminhos baseado em restrições fornecidas a partir do comportamento dos roteadores, de um OSS (*Operations Support System*), ou ainda de outro PCE da rede. A arquitetura PCE já possui um cálculo de caminho centralizado para grandes redes multidomínio e multicamadas, sendo adequada como ferramenta do SDN. Quando um nó necessita calcular o caminho para um determinado LSP, faz uma requisição ao PCE através do protocolo PCEP (*Path Computation Element Protocol*). O PCE tem acesso às informações de topologia de um domínio inteiro da rede. A arquitetura do PCE está ilustrada na Figura 5.22. Sua principal função é resolver o problema de multidomínios, pois o PCE tem a visão da rede como um todo, construindo uma base de dados através destes múltiplos domínios. A sessão entre o PCC (*Path Computation Client*) e o PCS (*PCE Server*) ou simplesmente PCE, é estabelecida sobre TCP, assim como o BGP. Uma vez a sessão estabelecida, o PCE constrói a base de dados de topologia TED (*Traffic Engineering Database*) usando um IGP como o OSPF ou o IS-IS, ou ainda através do BGP-LS. Este último possui estruturas TLV (*Type-Length-value*) que permitem ao PCE construir a base de dados. Quando o PCC solicita um LSP com certas restrições, o PCE calcula o caminho baseado na base de dados e responde com o caminho apropriado. Essa abordagem centralizada auxilia o cálculo e o provisionamento do caminho aumentando a flexibilidade e permitindo melhor uso dos recursos de rede. O PCE pode ser do tipo sem controle de estados (*Stateless PCE*) e com controle de estados (*Stateful PCE*), como ilustrado na Figura 5.23.

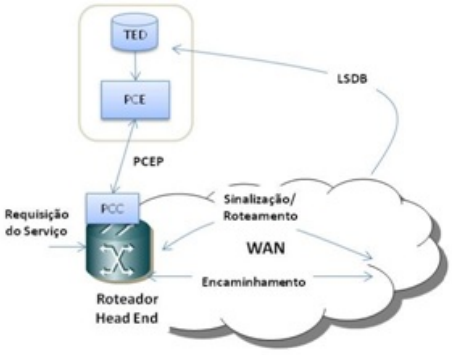


Figura 5.22. Arquitetura do PCE.

O PCE sem controle de estados tem habilidade reduzida para otimizar recursos de rede, não possui conhecimento prévio dos caminhos pré-estabelecidos. Ele é útil entre domínios do MPLS-TE, mas não é adequado para emprego em SDN. Já o PCE com controle de estados tem um controle do cálculo de caminho ótimo (por exemplo, estado

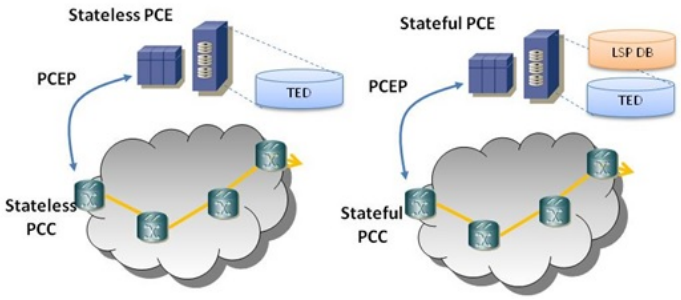


Figura 5.23. Stateless e Stateful PCE.

do LSP, recursos, políticas, análise de redes), possibilita a inicialização de caminhos e atualizações de controle, e requer sincronismo do estados da base de dados dos LSPs. O PCE com controle de estados pode ainda operar em modo passivo ou ativo. No modo passivo, o PCC inicia a configuração do caminho, atualizando as informações de controle, e o PCE aprende o estado do LSP para otimização do cálculo do caminho. No modo ativo, tanto o PCC quanto o PCE podem iniciar a configuração do caminho, mas a atualização do controle é feita pelo PCE, delegada pelo PCC. No modo ativo, o LSP pode ser iniciado pelo PCE, sendo mais integrado às demandas das aplicações, e o PCE pode fazer parte do controlador SDN determinando quando e quais caminhos serão configurados. No modo ativo, o LSP pode ser iniciado no PCC baseado nos estados que estão distribuídos na rede e pode ser usado em conjunto com os LSPs iniciados pelo PCE, sendo uma abordagem de rede híbrida com controle IP/MPLS e SDN [Paolucci et al., 2013]. A base de dados utilizada pelo PCE pode ser construída através de multidomínios, basicamente por quatro métodos distintos: cálculo de caminhos por domínio, cooperação entre PCEs, cálculo de caminhos de forma recursiva ou PCE hierárquico utilizado em multicamada IP/óptica.

Uma proposta de SDN para o PCE é uma solução ideal para operadoras de telecomunicações, desacoplando o plano de controle do plano de encaminhamento de dados, e com mecanismos de controle centralizado que controlam a configuração dos caminhos. Empregar o PCE com políticas apropriadas provenientes do OSS (*Operational Support Systems*), é a forma mais rápida e fácil para introduzir a SDN nas redes IP/MPLS e de transporte da operadora de telecomunicações. Com o emprego de PCEs, as redes de roteadores de núcleo têm um ganho substancial em benefícios de roteamento intra-domínios. Entende-se por domínio diferentes áreas de sistemas autônomos e também diferentes redes, como a rede IP e a de transporte óptica. Em redes legadas o PCE é implementado em servidores do sistema de gerência da rede, individualmente para cada domínio. A cooperação entre PCEs de diferentes domínios é uma questão de difícil solução técnica, e o SDN desponta como uma solução para o PCE multidomínios [Farrel, 2006].

5.3.3.3. Interfaces Northbound: REST, RESTFUL, NETCONF e outros

As interfaces *northbound* ainda são objeto de esforço de padronização e pesquisa para que garanta a interoperabilidade entre diferentes plataformas de controle. Alguns controladores como o Opendaylight e FloodLight propõem suas próprias NBIs. A API

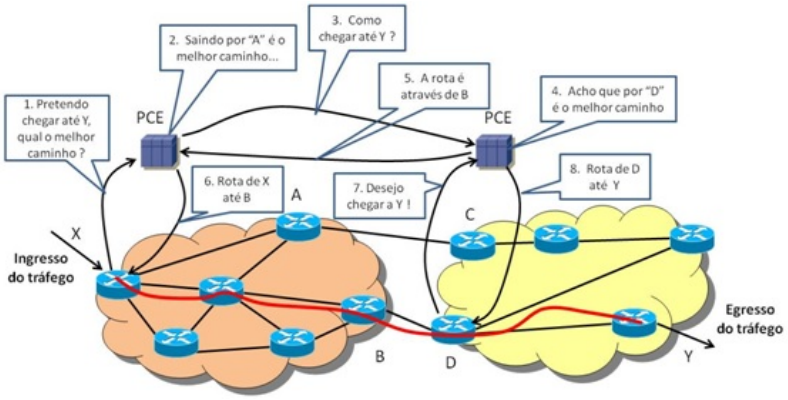


Figura 5.24. Exemplo de PCE intradomínios.

REST utiliza HTTP ou HTTPS, tendo sido desenvolvida inicialmente para acessar informações de serviços web. O uso do REST em SDN se deve a questões de simplicidade, flexibilidade, extensibilidade e segurança. Para acesso aos dados que envolvem recursos nos dispositivos de destino é simplesmente utilizada uma URL. A flexibilidade é consequência das entidades requisitantes poderem acessar os componentes de configuração definidas em um dispositivo usando os recursos REST em URLs diferentes. Não existem esquemas complexos ou MIBs para tornar o dado acessível. Já a extensibilidade do REST é devida ao suporte a novos recursos que não requerem a recompilação de esquemas ou MIBs, mas somente a chamada da URL apropriada pela aplicação. Por fim, usando o protocolo HTTPS, os aspectos de segurança são considerados e permite-se atravessar firewalls. A flexibilidade e extensibilidade podem representar um risco potencial do uso do REST pela falta de formalismo comparado a outros métodos. No entanto, a relação custo-benefício tende a indicar o REST como uma API propícia para utilização em SDN. A API REST pode trabalhar com diferentes formatos de arquivos como XML e JSON.

A NBI RESTCONF [Bierman et al., 2015] é um protocolo similar ao REST que roda sobre HTTP para acessar dados definidos em YANG e repositórios de dados em NETCONF. A RESTCONF descreve como mapear YANG em uma interface RESTful. A RESTCONF opera com repositórios de dados conceituais com a modelagem de dados YANG. O servidor lista cada módulo YANG suportado em um tipo de recurso da API de alto nível usando estruturas baseadas em módulos YANG habilitadas com URI. A RESTCONF pode enviar dados de requisição e resposta no formato JSON ou XML. O NETCONF possibilita a passagem de dados e comandos de e para dispositivos de rede, podendo ser usado interface SBI e NBI.

5.3.4. Plano de Gerenciamento

O plano de gerenciamento corresponde às aplicações utilizadas em Redes Definidas por Software e às linguagens de programação usadas para implementar estas aplicações. A subseção a seguir enumera algumas aplicações denominadas casos de uso em Redes Definidas por Software.

5.3.4.1. Aplicações de Rede

Entre as aplicações de rede destacam-se balanceadores de carga, lista de acesso para segurança, detecção de ataques, monitoramento da rede, virtualização da rede e roteamento. Neste último, se encaixa o Roteamento por Segmentos. A implementação da lógica de controle pode ser traduzida em comandos e instalados no plano de encaminhamento de dados ditando o comportamento dos dispositivos de encaminhamento.

- Engenharia de Tráfego: o principal objetivo desta aplicação é minimizar consumo de energia, maximizar a utilização agregada da rede, balanceamento de carga e outras técnicas de otimização de tráfego.
- Mobilidade e Redes Sem-fio: atualmente o plano de controle distribuído de redes sem-fio é subótimo face ao gerenciamento do espectro limitado, alocação de recursos de rádio, implementação de mecanismos de *handover* e balaceamento de tráfego entre células.
- Medição e Monitoramento: um exemplo de aplicações de medição é a melhoria da visibilidade do desempenho fim a fim da rede. As aplicações de monitoramento envolvem diferentes técnicas de amostragem e estimativas a serem aplicadas, reduzindo a interferência desnecessária do plano de controle nos dispositivos de encaminhamento, com o objetivo de coletar informações e calcular estatísticas do plano de dados.
- Segurança e Dependência: um conjunto de propostas diversas de segurança e dependência está emergindo no contexto de SDN. As vantagens do SDN para melhoria dos serviços depende de redes e sistemas seguros, como execução de políticas (controle de acesso,*firewall*, *middlebox*, a detecção e mitigação de ataques do tipo negação de serviço (*Denial of Service* - DoS), inspeção pormenorizada de pacotes (*Deep Packet Inspection* - DPI) e detecção de anomalias de tráfego. Existem basicamente duas abordagens de segurança: uma utilizando o SDN para prover segurança como serviço de valor adicionado, e outra, para provimento de segurança na própria arquitetura da SDN.
- Redes de Datacenter: as redes de centros de dados são beneficiadas de forma significativa solucionando problemas como migração de rede viva, gerência de rede avançada, implantação rápida do planejamento de redes para redes em produção. As aplicações em SDN caminham para um conceito de loja de aplicativos (SDN App Store), este conceito lançado pela HP, o controlador HP utiliza OpenFlow para acessar aplicações on-line e selecioná-las para serem dinamicamente descarregadas e instaladas no controlador. As linguagens de programação permitem prover um grande conjunto de poderosas abstrações [Casado et al., 2014] e mecanismos como composição de aplicações, tolerância a falhas no plano de dados e blocos de construção básicos. Um exemplo de linguagem de programação é o Pyretic que oferece uma abstração de alto nível da topologia e do conjunto de políticas da rede.

Ainda na visão simplificada da arquitetura de rede SDN, a plataforma de controle se comunica com as aplicações de rede através da interface *northbound*, que oferece uma API

para desenvolvedores de aplicação, como a API REST, RESTCONF, Restful e linguagens de programação com Java, Python, C e C++ [Hodzic e Zoric, 2008].

5.3.4.2. Roteamento como serviço: Routeflow e RCP

A utilização de SDNs para decisão de encaminhamento [Adrian et al., 2015], como o Roteamento por Segmentos, é uma tendência como por exemplo o RCP e RouteFlow. O RCP (*Routing Control Plataform*) é uma alternativa primitiva de roteamento como serviço [Feamster et al., 2004]. O servidor RCP divulga as rotas aos roteadores usando protocolos existentes como o BGP. A visão da rede é obtida pelo IGP. O RPC opera de forma distribuída, dividindo a o projeto em componentes, visualizando o estado global da rede por componente. O RCP possui objetivos similares aos das redes SDN, tais como melhor escalabilidade, gerenciamento e separação do software do hardware do roteador. O projeto RouteFlow é considerado um caso de uso de implementação de uma rede de teste para roteamento como serviço (*IP Routing-as-a-Service*) em código aberto, demonstrando a migração de uma rede tradicional de camada 3 para uma rede OpenFlow de camada 3. Os equipamentos roteadores legados interoperam com implementações simplificadas de roteamento intra e interdomínio. O projeto RouteFlow [Rothenberg et al., 2011] foi particularmente importante pois demonstrou um método de integração de redes tradicionais com inúmeros protocolos de rede com redes definidas por software baseada em OpenFlow [Jingjing et al., 2014]. Os conceitos de SDN são importantes para a compreensão do roteamento por segmentos, visto que o roteamento por segmentos é uma aplicação SDN.

5.4. Roteamento por Segmentos

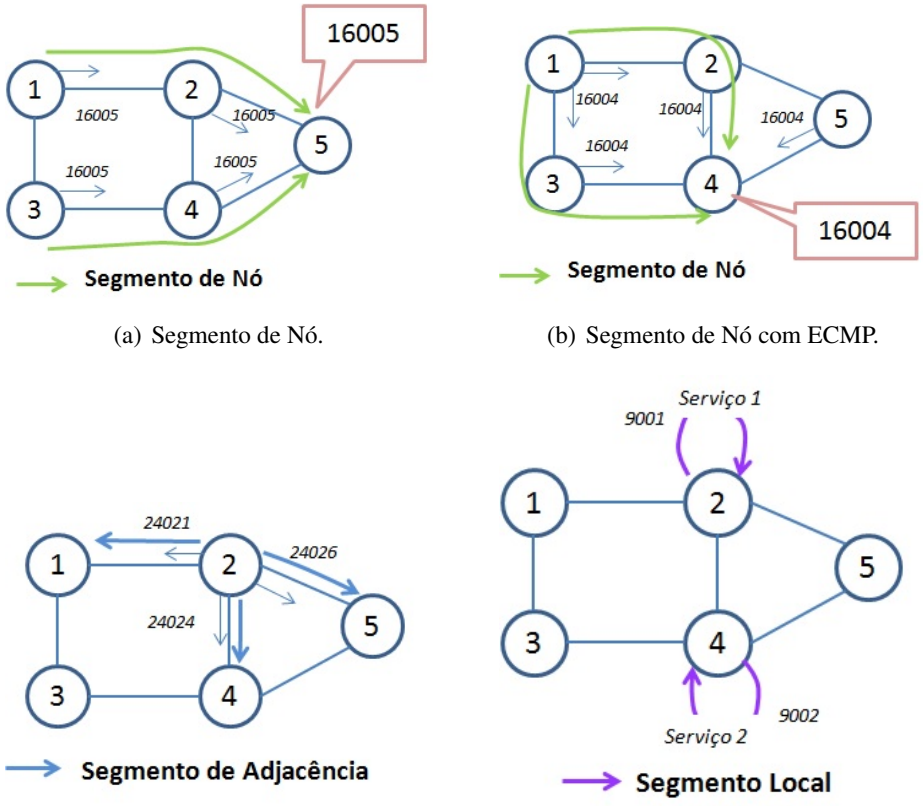
O Roteamento por Segmentos ou SR (*Segment Routing*) é um conceito que pode ser entendido como um protocolo de roteamento suportado por aplicações de Redes Definidas por Software para a definição de caminhos de forma eficiente e automatizada. O Roteamento por Segmentos melhora o encaminhamento de pacotes sem necessitar da manutenção de estados por fluxo dentro da rede de roteadores de núcleo, reduzindo a complexidade dos planos de controle e de dados. A engenharia de tráfego é um exemplo de aplicação no contexto de Roteamento por Segmentos. O Roteamento por Segmentos permite a configuração, a modificação e a remoção de caminhos TE dentro de um domínio de rede, operando somente na borda da rede. O plano de controle de Roteamento por Segmentos pode ser mantido de forma centralizada ou distribuída.

5.4.1. Conceitos preliminares

O Roteamento por Segmentos define um protocolo de roteamento pela origem, ou seja, a origem do fluxo de dados escolhe e codifica no cabeçalho do pacote a lista de segmentos a serem percorridos pelos pacotes até o destino. O segmento, por sua vez, é um identificador genérico para qualquer tipo de instrução: um serviço, contexto, localizador ou um caminho baseado no IGP ou no BGP. Ainda, um segmento pode ser representado por um índice local ou global. Por exemplo, uma lista de segmentos para a rede de roteadores de núcleo IP/MPLS é representada por uma pilha de rótulos (*Label Stack*), enquanto no IPv6 é representada pela extensão do cabeçalho para rotea-

mento [Previdi et al., 2015b]. O identificador de segmento denomina-se SID (*Segment Routing Identifier*), e em redes MPLS, o SID é um rótulo MPLS. A cadeia de SIDs é chamada de caminho de Roteamento por Segmentos (*Segment Routing Path - SR path*).

Os segmentos podem ser divididos em Segmentos de Nó, Segmentos de Adjacência e Local. A Figura 5.25 mostra o identificador de Segmento de Nó, de Adjacência e Local. O Segmento de Nó tem uma numeração única e global denominada prefixo SID retirado do SRGB (*Segment Routing Global Block*) [SPRING, 2015] dentro do domínio do IGP, que também corresponde a um domínio do Roteamento por Segmentos. Os Segmentos de Nó estão contidos em uma numeração global de segmentos. A numeração global de segmentos, em redes MPLS, é formada por uma faixa de rótulos reservada, somada a um índice referente ao nó (prefixo), compondo a numeração do Segmento de Nó. O Segmento de Adjacência tem significado local e está relacionado a uma ou mais adjacências do nó. Os Segmentos de Adjacência têm rótulo no formato 240xy para uma determinada adjacência xy. Os rótulos de 90000 a 99999 são usados pelas extensões dos protocolos LDP, RSVP e BGP. O Segmento Local corresponde ao segmento suportado apenas no nó que foi gerado. Assim, nenhum outro nó pode instalar este SID em sua FIB.



(a) Segmento de Nó.

(b) Segmento de Nó com ECMP.

(c) Segmento de Adjacência.

(d) Segmento Local.

Figura 5.25. Segmento de Nó, Adjacências e Local.

Na Figura 5.25(a), o rótulo 16005 corresponde ao Segmento de Nó 5. O identificador do segmento é composto pelo SRGB no valor de 16000 a ser detalhado na Subseção 5.4.3 e mais um valor de um prefixo do nó, neste caso 5. Nota-se que o SID 16005

é o mesmo para qualquer nó de origem, a partir de qualquer nó de origem utiliza-se o Segmento de Nó 16005 para chegar ao nó 5 de destino e a divulgação do SID é feita pelo IGP para todos os nós. Na Figura 5.25(b), existem dois caminhos de mesmo custo entre o nó 1 e 4, obtidos pelo IGP. A escolha do caminho depende de outras restrições como latência e banda, obtida por uma aplicação SDN que possua visão completa da rede. A Figura 5.25(c) ilustra os Segmentos de Adjacências do nó 2 para o nó 1, 4 e 5 com SIDs 24021, 24024 e 24025, respectivamente, e representa uma identificação para cada enlace diretamente conectado ao nó. Neste exemplo, a numeração escolhida para os Segmentos de Adjacência foi 240xy (faixa de identificador de segmento fora do SRGB que identifica o Segmento de Nó conforme descrito na Subseção 5.4.3). Para cada Segmento de Adjacência foi utilizado o identificador xy, onde xy é uma adjacência que identifica o enlace entre os nós x e y. A Figura 5.25(d) exemplifica um Segmento Local no nó 2 de SID 9001, indicando um serviço existente no nó 2. Em particular para o Segmento Local, nenhum outro nó pode configurar este SID na SR-FIB (*Segment Routing-Fowarding Information Base*) sob pena de conflito de SIDs. Os segmentos de Nó e de Adjacência podem ser combinados, dirigindo o tráfego através de qualquer caminho na rede.

O caminho é especificado como uma lista de segmentos no cabeçalho do pacote através de um empilhamento de rótulos MPLS (Figura 5.26). Não existem estados por fluxo nos roteadores intermediários, utilizando-se apenas o IGP, e consequentemente, dispensando o protocolo de distribuição de rótulos (LDP) de uma rede MPLS convencional. Os estados por fluxo são mantidos apenas na origem, pois o roteador de origem conhece todos os SIDs para alcançar o destino. A quantidade de entradas na tabela SR-FIB em um determinado nó é da ordem de $N + A$, onde N é o é numero de Segmentos de Nó e A é o número de Segmentos de Adjacências, como observada na Figura 5.26. Ainda na Figura 5.26, a informação do Segmento de Nó 4 com SID de 16004 é anunciada aos demais roteadores pelo IGP, correlacionando o SID com a interface loopback do nó 4. O Roteamento por Segmentos possui operações similares ao das redes IP/MPLS. O segmento ativo é definido como o segmento que vai aplicar a instrução corrente no pacote. Nas redes MPLS, o SID do segmento ativo é o rótulo mais externo; enquanto no IPv6, é um ponteiro para o SID. No exemplo da Figura 5.26, na interface de egresso do nó 1 em direção ao nó 2, o segmento ativo é o com SID 16004, que é o Segmento de Nó que vai encaminhar o pacote até o nó 4, compondo uma parte do caminho. As operações que podem ser efetuadas em Roteamento por Segmentos são as seguintes:

- `PUSH`: Inserção em uma lista de segmentos. Para redes MPLS, significa inserção do rótulo na pilha; enquanto para o IPv6, significa inserir o SID na primeira posição e redirecionar o ponteiro o topo da lista.

- `NEXT`: Ativação do próximo segmento da lista, uma vez que o atual está completo. Nas redes MPLS, significa remover o rótulo mais externo; enquanto para o IPv6 significa incrementar o ponteiro.

- `CONTINUE`: Continuação do atual segmento ativo, mesmo sem estar completo. Nas redes MPLS, há correspondência com o processo de comutação de rótulos. Logo, se o próximo salto está no mesmo SRGB, o valor do rótulo é mantido. No IPv6, significa não incrementar o ponteiro.

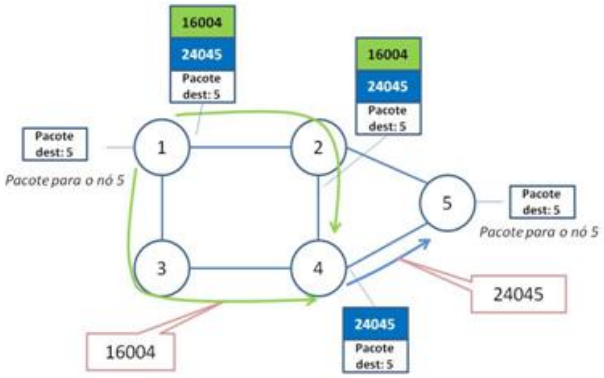


Figura 5.26. Combinação de Segmentos de Nó e de Adjacências.

A Figura 5.27 mostra as operações `PUSH` (Insere), `NEXT` (Próximo) e `CONTINUE` (Continue) implementadas no Roteamento por Segmentos referente ao exemplo da Figura 5.26. O segmento ativo é aquele cujo SID corresponde ao rótulo mais externo, neste caso nos enlaces 1-2, 2-3 o SID ativo é do Segmento de Nó global 16004. Nota-se que no nó 2, o rótulo passa por um processo de comutação MPLS, sem no entanto, alterar o SID. No nó 4, o rótulo 16004 é removido e o segmento ativo passa a ser o Segmento de Adjacência com SID 24045. Finalmente, no nó 5 é feita a remoção do rótulo atual mais externo com SID 24045, entregando o pacote à rede de destino.

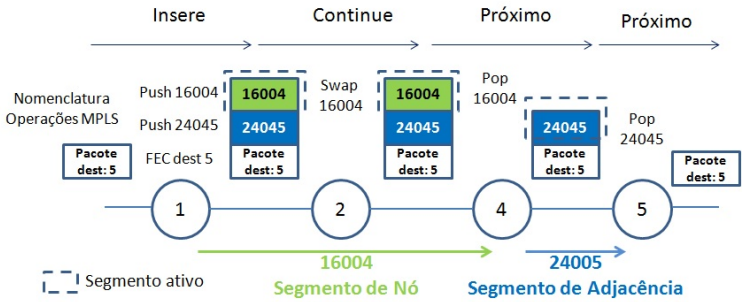


Figura 5.27. Operações em Roteamento por Segmentos.

Uma boa prática para a Operadora de Telecomunicações é alocar o mesmo SRGB em um domínio de Roteamento por Segmentos (*SR-Domain*). Um SR-Domain é um conjunto de nós conectados em uma infraestrutura física correspondente a uma rede de Operadora de Telecomunicações, e confinada dentro de uma instância do IGP, chamada SR-IGP (*Segment Routing IGP*). Por exemplo, uma rede típica de 500 nós (500 Segmentos de Adjacência) com 5000 segmentos de nó globais (pertencentes ao SRGB), para um determinado fluxo f, somente o nó de ingresso do tráfego detém os estados para f, ou seja 5500 entradas na SR-FIB. Mesmo que se tenha $n \times f$ fluxos, a quantidade de entradas na SR-FIB é a mesma. Outra facilidade do Roteamento por Segmentos é o agrupamento (*bundle*) através de Segmentos de Adjacência. Os Segmentos de Adjacências permitem

agrupar múltiplos enlaces físicos, permitindo balanceamento de tráfego, ou configurar o SID para um enlace específico da adjacência. Por exemplo, na Figura 5.28, os Segmentos de Adjacências 24045, 24145 são configurados em enlaces físicos distintos, na mesma adjacência, enquanto o SID 24245 representa um grupo (*bundle*) de enlaces físicos, permitindo balancear o tráfego de forma simples.

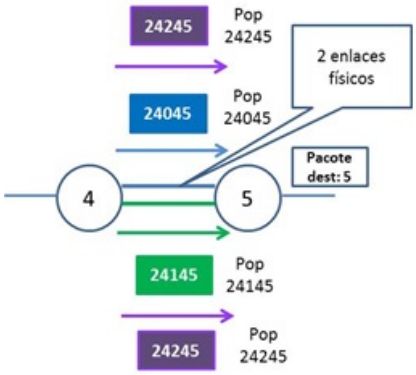


Figura 5.28. Segmentos de Adjacência (*bundle*).

O Roteamento por Segmentos tem foco no plano de encaminhamento de dados e a inteligência da escolha do melhor caminho é realizada pela Rede Definida por Software. O Roteamento por Segmentos simplifica a operação da rede da Operadora de Telecomunicações, reduzindo a quantidade de protocolos de controle e estados nos roteadores intermediários do núcleo da rede e, consequentemente, permitindo a criação de serviços baseado em aplicações através do SDN. O roteamento aumenta a capacidade da rede, aproveitando melhor a infraestrutura existente e evitando a duplicação de túneis TE para salvamento do tráfego. Comutando em sub-50ms, o uso do Roteamento por Segmentos em redes IP/MPLS permite uma vasta gama de aplicações para Operadoras de Telecomunicações e Operadoras OTT (*Over The Top*) através de redes de acesso banda larga e centro de dados. A Tabela 5.1 mostra as diferenças entre a arquitetura IP/MPLS e a arquitetura das redes que utilizam Roteamento por Segmentos (*SR-based MPLS*), destacando as vantagens da arquitetura.

5.4.2. Plano de dados: MPLS e IPv6

Em redes MPLS, o Roteamento por Segmentos utiliza o plano de encaminhamento de dados do MPLS, onde o segmento é representado por um rótulo; e a lista de segmentos, é representado pelo empilhamento de rótulos. O Roteamento por Segmentos possui as funcionalidades de PHP (*Penultimate Hop Popping*) e rótulos *explicit-null*. A imposição do rótulo representa o prefixo do SID e tem preferência em casos do prefixo de destino não possuir um rótulo associado através do protocolo LDP. O encaminhamento do pacote e as operações de Roteamento por Segmentos são ilustrados na Figura 5.29. O nó 4 utiliza prefixo IPv4 ou IPv6 de sua interface loopback:1.1.1.4/32, com prefixo de SID associado 16004, assumindo que o protocolo LDP não esteja habilitado. O nó 4 requisita a funcionalidade PHP por padrão. Esse prefixo é uma entrada na FIB do roteador remoto 1 e a operação executada é a inserção do rótulo (*push*). O prefixo SID também é uma

Tabela 5.1. MPLS com Roteamento por Segmentos *vs.* MPLS tradicional.

Característica	Rede MPLS com SR	Rede MPLS tradicional
Transporte MPLS básico	IGP	IGP+LDP
Sincronismo entre o LDP/IGP	Não se aplica	Difícil de gerenciar
Comutação FRR em 50ms	IGP	IGP+LDP
Túneis TE adicionais para suportar FRR	Não precisa	Precisa
Otimização de caminho de backup	Sim	Não
ECMP para criação de túneis TE	Sim	Não
Estados apenas no *head-end* do TE	Sim	Não, dependendo do número de nós (complexidade $O(n^2)$ nos pontos intermediários)
Interoperabilidade com a rede MPLS tradicional	Sim	Não se aplica
Projetado para SDN	Sim	Não

entrada da LFIB do roteador 2, com a operação de comutação do rótulo (*swap*). No roteador 3, o prefixo SID é uma entrada remota de sua LFIB. Como o roteador 3 é o penúltimo salto (PHP), então a operação executada sob o rótulo é de remoção (*pop*). No roteador 4, o pacote chega sem o rótulo baseado no endereço IP ou o rótulo do serviço (Segmento Local).

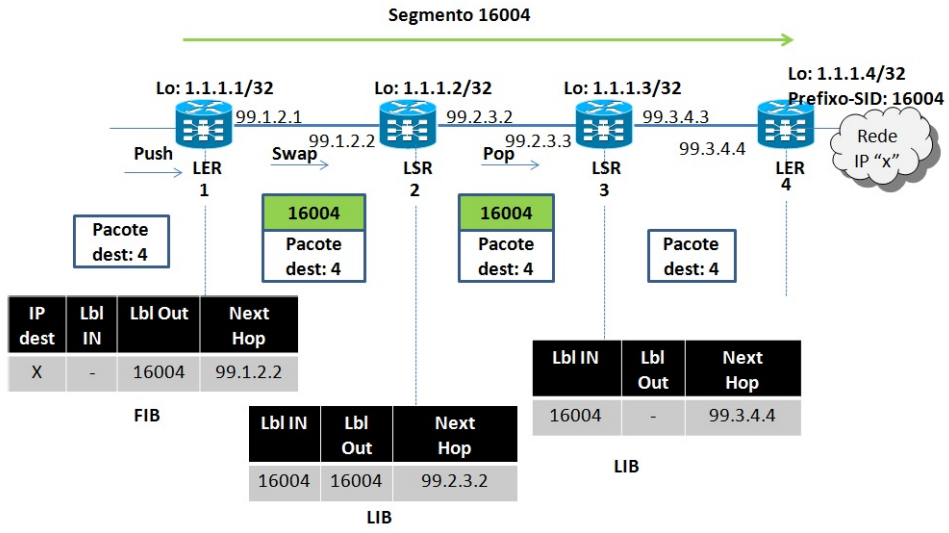


Figura 5.29. Roteamento de Segmentos - Plano de dados MPLS.

A opção de *explicit-null* é configurável para um prefixo SID, sendo que o vizinho do roteador de origem do SID comuta o prefixo SID mais externo com o rótulo *explicit-null*. Na Figura 5.29, a LIB do roteador 3, o rótulo de saída (*Label Out*) seria *explicit-null*, sendo útil para repassar os EXP bits do LSR 3 para o LER 4.

O plano de dados MPLS em Roteamento por Segmentos é o mesmo das redes IP/MPLS legadas. Para serviços VPN L3, a rede MPLS se torna um serviço de transporte

baseado nos prefixos de segmentos. A Figura 5.30 ilustra um serviço VPN L3 sobre uma rede implementada com Roteamento por Segmentos. Os prefixos SID correspondem aos Segmentos de Nó, que são SIDs globais. Nota-se que para os nós adjacentes ao nó 3, os SID globais são removidos. Para o nó 6 existem duas possibilidades de caminho de custo igual (ECMP) com mesmo SID global. Do nó 3 para seus vizinhos (nó 1 e 4) são configurados Segmentos de Adjacência com significado local, sendo removidos.

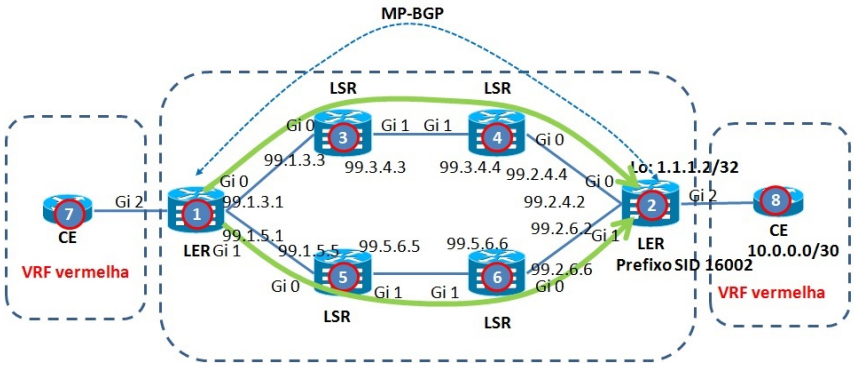


Figura 5.30. VPN L3 com Roteamento por Segmentos.

Além das redes MPLS, o Roteamento por Segmentos pode ser aplicado no plano de dados IPv6, onde a lista de segmentos está codificada na extensão do cabeçalho para roteamento pela fonte. A lista de segmentos pode ser baseada no IGP ou no BGP. A divulgação dos segmentos na rede com IPv6 é feita a partir de extensões dos protocolos IGP, BGP, BGP-LS e PCEP. O Roteamento por Segmentos em IPv6 não necessita de atualização de toda a rede, permitindo interoperar com e sem Roteamento por Segmentos. Essa característica possibilita a adoção gradual da tecnologia. O caminho é expresso de forma explícita, onde os nós representam roteadores, servidores, instâncias de aplicações, serviços, cadeias de serviços, etc. O roteamento em segmentos em IPv6 é um roteamento pela origem não-estrito.

Figura 5.31. Cabeçalho de Roteamento por Segmentos SRH do IPv6.

No cabeçalho do IPv6 (*Segment Routing Header* - SRH) visto na Figura 5.31, os campos de "Segment List", descrevem o caminho do pacote. O segmento é representado

por um endereço IPv6. A seguir são detalhados os campos do cabeçalho de Roteamento por Segmentos:

- `Next Header`: é um seletor de 8 bits que identifica o tipo de cabeçalho imediatamente seguido pelo SRH.
- `Hdr Ext Len`: define o tamanho do cabeçalho de SRH em octectos, descontando os primeiro oito octetos.
- `Routing Type`: ainda depende de definição pelo IETF.
- `Segments Left`: contém o índice da lista de segmentos, indicando o próximo a ser inspecionado. É decrementado a cada inspeção.
- `First Segment`: é um "offset" no SRH, não incluindo os 8 primeiros octetos. É expresso em múltiplos de 16 octetos apontando para o último elemento da lista de segmentos. Representa o primeiro segmento da lista.
- `Flags`: 16 bits para flags. Os bits de 4 a 15 definem o tipo de codificação dos endereços IPv6 na lista de políticas (Policy List).
- `Segment List[n]`: é o endereço IPv6 que representa cada segmento do caminho. A lista de segmentos é codificada de forma reversa, ou seja, o último segmento é o primeiro da lista.
- `Policy List[n]`: são endereços que representam nós específicos no SR-Path: "Ingress SR PE" (nó que insere o cabeçalho SRH) e "Egress SR PE" (endereço do nó de egresso do domínio de Roteamento por Segmentos).
- `HMAC`: autenticação SRH, ainda em versão "draft" pelo IETF.

O SRH é um novo tipo em um cabeçalho de roteamento existente no IPv6, idêntico ao RH0, que tornou-se obsoleto por questões de segurança. O SRH utiliza o HMAC como solução de segurança usado no ingresso de um domínio de Roteamento por Segmentos segundo o "draft" do IETF *draft-vyncke-6man-segment-routing-security*. Dentro de um domínio controlado de Roteamento por Segmentos, o HMAC não é necessário. No Roteamento por Segmentos do IPv6, o segmento ativo é aquele que está designado como endereço MAC de destino no pacote. Em cada ponto final do segmento (*endpoint*), o endereço MAC de destino é atualizado com o próximo segmento ativo na lista de segmentos conforme a Figura 5.32. Na topologia apresentada na mesma figura, o nó A é chamado nó de Roteamento por Segmentos habilitado (*SR Capable*), capaz de criar a lista de segmentos ou recebê-la de um controlador SDN. Os nós intermediários do caminho são chamados de nós de trânsito (*Transit Nodes*), que podem não executar o Roteamento por Segmentos, nós com Roteamento por Segmentos intra-segmento (*Intra-Segment Nodes*) e nós finais (*Endpoint Nodes*). Os nós B e G são nós de trânsito sem Roteamento por Segmentos, fazendo o encaminhamento dos pacotes baseados no endereço IPv6 sem inspecionar o SRH. Se ocorresse a inspeção do SRH, esses nós seriam chamados de nós intra-segmentos. Os nós C, F e H são nós finais do Roteamento por Segmentos, pois inspecionam o SRH, e também são nós de Roteamento por Segmentos habilitados.

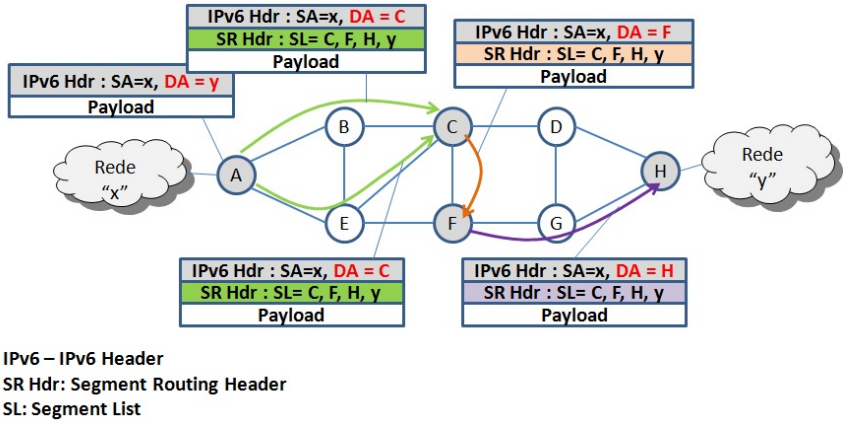
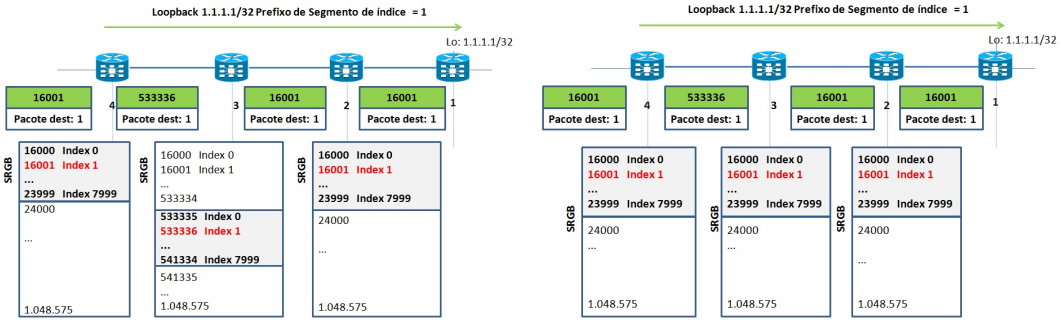


Figura 5.32. Exemplo de plano de dados IPv6 para Roteamento de segmentos.

5.4.3. Bloco Global de Roteamento por Segmentos (SRGB)

O Bloco Global de Roteamento por Segmentos (*Segment Routing Global Block* - SRGB) é uma faixa de rótulos reservada para os Segmentos de Nó ou globais, pois tem valor absoluto em um domínio de Roteamento por Segmentos. O prefixo SID é anunciado com um índice único em um domínio de Roteamento por Segmentos. O primeiro prefixo começa em zero, e o rótulo é formado pelo prefixo do SID somado à base do SRGB, representando o Segmento do Nó. Por exemplo, um roteador com interface loopback 1.1.1.65/32 tem prefixo SID 65 com rótulo 16065. Uma boa prática é usar o mesmo SRGB em todos os nós do mesmo domínio de Roteamento por Segmentos, pois diferentes SRGBs podem complicar a solução de problemas na rede. O SRGB não padrão pode ser alocado com rótulos entre 16000 a 1048575 ou até onde o roteador permitir, sendo o tamanho máximo de 64 kBytes. A Figura 5.33(a) mostra uma possível alocação de SRGB não recomendada, com SRGBs diferentes. Os SRGBs diferentes podem ter conflito com rótulos distribuídos pelo LDP em uma rede mista com Roteamento por Segmentos e IP/MPLS legada (Subseção 5.4.5). A Figura 5.33(b) mostra a alocação recomendada com o mesmo SRGB, o que simplifica a programação na rede SDN.



(a) Não recomendada. (b) Recomendada.

Figura 5.33. Alocação de SRGBs.

Os rótulos que representam os SIDs globais e locais são gerenciados por uma base

de comutação de rótulos (*Label Switching Database* - LSD) que aloca dinamicamente rótulos para as aplicações do MPLS, tais como os protocolos LDP, RSVP, BGP, TE, redes virtuais privadas de camada 2 e os Segmentos de Adjacência obtidos pelo IGP. O LSD preserva a faixa de rótulos do SRGB (de 16000 a 23999) e aloca dinamicamente os rótulos a partir de 24000. O LSD pode alocar rótulos dinamicamente do SRGB em situações de emergência, quando a faixa de rótulos dinâmicos foi consumida, ou se a faixa reservada para o SRGB não for usada. A LSD permite que futuras ativações de Roteamento por Segmentos em roteadores de núcleo não necessitem de um "reboot", coexistindo com rótulos já alocados pela rede IP/MPLS legada. No primeiro momento da ativação do LSD, este aguarda que o IGP solicite o SRGB, assim o LSD aloca a faixa de rótulos do SRGB, e o IGP já pode usá-lo. A Tabela 5.2 mostra a alocação de rótulos para o MPLS e para o Roteamento por Segmentos.

Tabela 5.2. Faixa de rótulos do MPLS e do Roteamento por Segmentos.

Uso	Faixa de Rótulos
Reservada para uso especial	0 a 15
Reservada para rótulos MPLS estáticos	16 a 1599
Reservada para Roteamento por Segmentos	16000 a 23999
Reservada para roteamento dinâmico	24000 a 1048575

5.4.4. Plano de controle IGP do Roteamento por Segmentos: IS-IS e OSPF

O plano de controle IGP é responsável pela configuração e distribuição dos segmentos, aplicando os segmentos em redes multi-área (OSPF) e multinível (IS-IS) e verificando os anúncios de rotas, através de extensões desses protocolos. O IS-IS suporta o plano de controle IPv4 e IPv6, com roteamento considerando múltiplos níveis de rede (nível 1 e 2 do IS-IS). Utiliza o prefixo do SID para representar as interfaces loopback dos roteadores em IPv4 e IPv6 e utiliza os Segmentos de Adjacência para identificar as adjacências dos nós. O anúncio do prefixo para o SID é feito pelo servidor de mapeamento (*Mapping Server*). O Roteamento por Segmentos com IS-IS torna possível a introdução do suporte de sub-TLVs (*Type-length-value*) em extensões do protocolo IS-IS [Previdi et al., 2015a]. Para o OSPF, a versão que suporta as extensões para Roteamento por Segmentos é o OSPFv2, com multi-área, no qual o prefixo de SID representa as interfaces loopback dos roteadores em IPv4 e os Segmentos de Adjacência identificam as adjacências do nó conforme o "draft" do IETF *draft-ietf-ospf-segment-routing-extensions-05*. As extensões do OSPF para Roteamento por Segmentos adicionam anúncios de estado de enlace opaco (*Opaque LSA*), que permitem a transmissão arbitrária de dados que o OSPF não necessariamente utilize. Os anúncios de estados de enlace opaco adicionados oferecem suporte ao Roteamento por Segmentos, permitindo o envio de informações como o algoritmo usado no roteamento e as faixas de rótulos (*Opaque LSA Type 4*), SID de Segmentos de Nó (*Opaque LSA Type 7*) e SID de Segmentos de Adjacência (*Opaque LSA Type 8*). O prefixo do SID do Segmento do Nó usa a informação do SRGB, que á anunciado pelos LSAs opacos. O SRGB pode ser o padrão que utiliza rótulos MPLS de 16000 a 23999 ou algum outro que utilize rótulos fora do padrão. O SRGB escolhido pode ser configurado em cada instância do IGP, sendo que, as diferentes instâncias podem usar SRGB iguais (*Overlapping SRGB*) ou diferentes (*Non-overlapping SRGBs*).

O Prefixo do SID pode ser configurado como um valor absoluto ou índice, sendo que o índice representa um incremento na base do SRGB. Por exemplo, considerando um prefixo com índice igual a 1 e um SRGB igual a 16000, então o SID é $16000 + 1 = 16001$. Esse valor de SID representa o rótulo, tem valor global e é único em um SR-Domain. O Prefix SID é configurado manualmente e equivale a atribuir um endereço a uma interface loopback do roteador de núcleo. Por exemplo, o roteador originador pode anunciar as faixas de rótulos $[100, 199]$, $[1000, 1099]$ e $[500, 599]$ fora do SGRB padrão. Nesse caso, os roteadores de destino que receberem essas faixas, as concatenam formando a SRGB $\{[100, 199], [500, 599], [1000, 1099]\}$. Dessa forma, os índices utilizam várias faixas, como por exemplo: Índice 0 = Rótulo 100, Índice 1 = Rótulo 101, ..., Índice 99 = Rótulo 199, Índice 200 = Rótulo 500, Índice 201 = Rótulo 501, ..., Índice 299 = Rótulo 599, Índice 300 = Rótulo 1000, Índice 301 = Rótulo 1001, ..., Índice 399 = Rótulo 1099.

O Segmento de Adjacência tem significado local e é automaticamente alocado para cada adjacência, sempre sendo um valor absoluto não indexado. As extensões do OSPF possuem TLVs e sub-TLVs com flags para o anúncio e propagação dos prefixos. A Figura 5.34 ilustra as extensões do OSPF e respectivos flags para suporte aos SIDs.

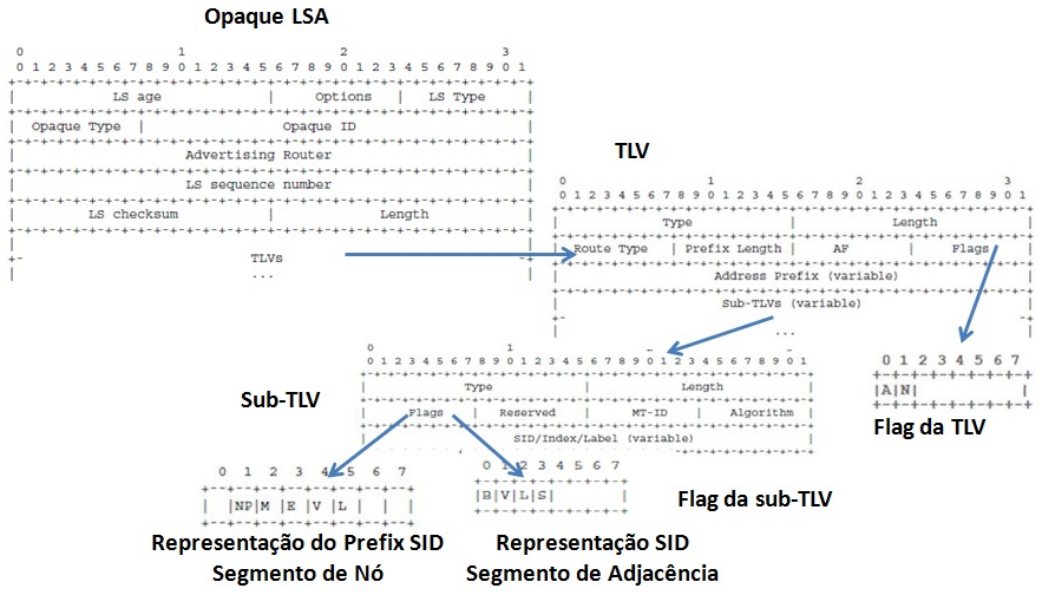


Figura 5.34. Extensões do OSPF para prefixos SID.

O campo de flags da TLV é composto por dois bits, sendo que o bit A sinaliza o uso de uma extensão TLV para prefixo SID inter-área e o bit N indica que o prefixo identifica o nó. No campo de flags da sub-TLV, o bit NP configura o PHP, o que significa que o prefixo SID não pode ser removido depois do encaminhamento do pacote; o bit M indica se os SIDs são anunciados pelo servidor de mapeamento; o bit E indica que o penúltimo salto deve trocar o prefixo SID por um rótulo *explicit-null*, o bit V indica que o prefixo SID é um valor absoluto e, finalmente, o bit L indica quando o prefixo SID tem significado local, ou seja, é um índice.

Existem Prefixos SID denominados de *Anycast* por serem anunciados a múltiplos

nós da rede. O tráfego é encaminhado pelo originador do prefixo SID *Anycast*, escolhendo a melhor rota baseada no IGP. Os nós que anunciam o mesmo prefixo SID *Anycast* tem o mesmo SGRB. Esses prefixos são divulgados entre diferentes áreas através dos roteadores ASBR (*Autonomous System Boundary Router*), responsáveis por distribuir rotas de outros roteadores de outras áreas e de outros sistemas autônomos.

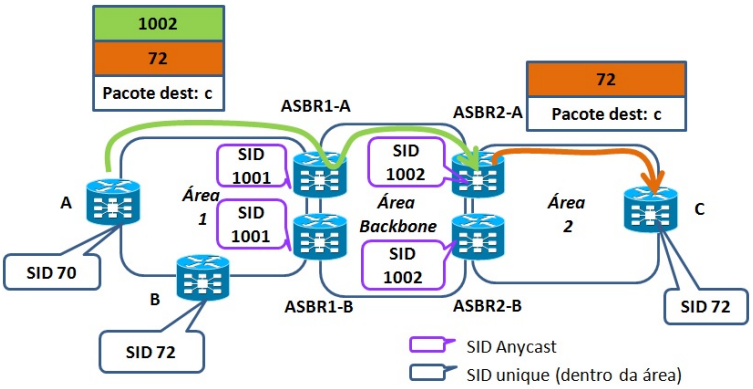


Figura 5.35. Prefixo SID *Anycast*.

A Figura 5.35 mostra um exemplo de aplicação de prefixo SID *Anycast*. Em redes multi-áreas, os SIDs dos ASBRs são *Anycast*, pois são únicos em todo domínio e são redistribuídos em cada região. Apesar dos SIDs serem únicos internamente nas áreas não-backbone, eles podem ser reutilizados em outras áreas. Por exemplo, o rótulo 72 leva até C dentro da área 2 e B dentro da área 1. Porém, os rótulos {1001,72} levam até B de qualquer lugar, e {1002,72} levam até C de qualquer lugar. Os rótulos 1001 e 1002 são divulgados em todo o domínio SR, possuindo re-roteamento rápido nativo (*Fast Reroute* - FRR) quando ocorrer falha em algum dos ASBRs. No OSPF, quando um nó anuncia seu prefixo SID, ele inclui este prefixo nas *Opaque LSAs* que são enviadas a seus vizinhos. Ao propagar um prefixo, o OSPF anuncia um prefixo recebido ou originado de outra área. Quando um nó cria um prefixo, anunciando-o como local (Segmento Local), o nó é proprietário deste prefixo. No OSPF, os prefixos SID são propagados entre áreas, enquanto os de adjacências não são. O OSPF não tem como identificar quais nós originaram o prefixo e quais propagaram este prefixo. As flags também transportam informações do comportamento do nó originador do prefixo, como o *explicit-null*. Esse comportamento não deve ser aplicado aos nós propagadores do prefixo, mas apenas ao originador. Em multi-áreas OSPF, o ABR (*Area Border Router*) ou ASBR propaga o prefixo SID não-locais com o bit NP=0 (*No PHP*) e o bit E=0 (*no-explicit-null*). Já para prefixos locais, o prefixo SID é propagado com bit NP=1 e o bit E=0. A Figura 5.36 mostra um exemplo de multi-área OSPF com prefixos SID locais e não-locais que atravessam as áreas OSPF.

Os Segmentos de Adjacências tem significado local e são alocados dinamicamente a partir de um conjunto de rótulos conforme mostrado na Figura 5.28. A alocação automática de rótulos pode ser feita por adjacência, como por exemplo uma adjacência com proteção e sem proteção. No IS-IS, isso significa ter dois SIDs diferentes para adjacências L1 e L2 entre os mesmos vizinhos; no OSPF, o mesmo SID de adjacência é aplicado em todas as áreas de uma adjacência multi-área, o que representa múltiplas adjacências para

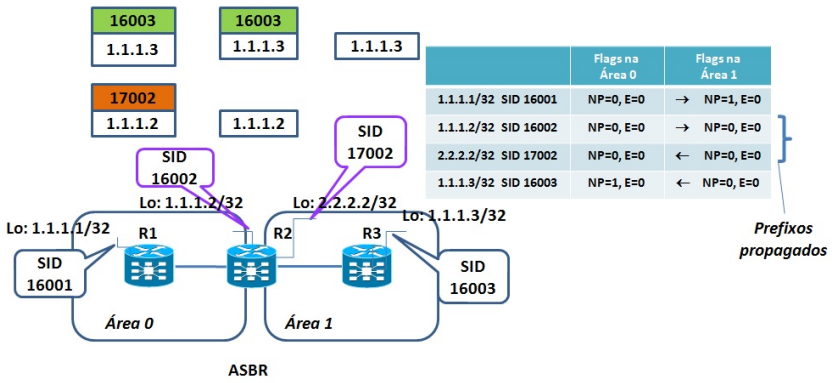


Figura 5.36. Propagação de prefixos SID e flags.

cada área diferente, em uma mesma interface. Os Segmentos de Adjacência têm persistência de rótulo, ou seja, o mesmo rótulo é alocado depois da recuperação de uma falha. No OSPF, os Segmentos de Adjacências são representados nas "flags" sub-TLVs do *Opaque LSA*, com os seguintes bits quando configurado com "1": B é o bit de Backup, que indica uma adjacência protegida; o bit V significa que o SID de adjacência transporta um valor e não um índice; o bit L significa que o SID de adjacência tem significado local e o bit S quando o SID de adjacência se refere a um conjunto de adjacências, usado para balanceamento de carga conforme mostrado na Figura 5.36. No Roteamento por Segmentos, os nós que estejam interligados através de uma rede local necessitam conduzir os fluxos de dados nesta rede. Para tal, os nós necessitam de Segmentos de Adjacência através da rede local. A solução é a criação de um "pseudo-nó"representando a rede LAN. Os Segmentos de Adjacência são associados aos nós ligados à LAN e ao pseudo-nó (Figura 5.37).

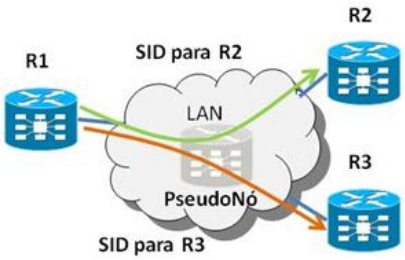
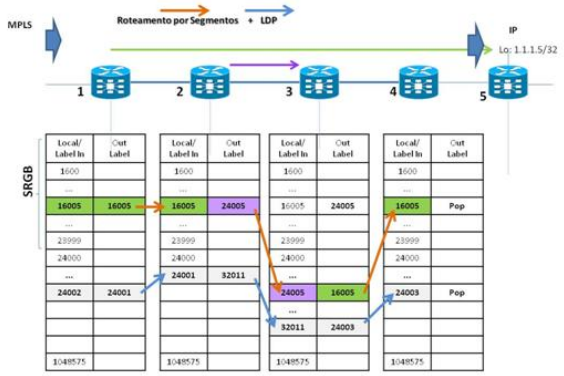


Figura 5.37. Segmento de Adjacências - Pseudo-nó.

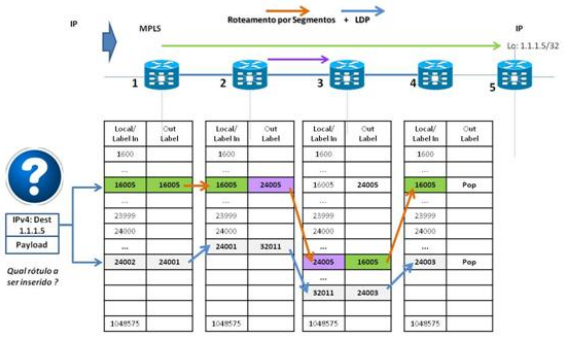
5.4.5. Coexistência do Roteamento por Segmentos e LDP

A arquitetura de redes IP/MPLS permite a coexistência de múltiplos protocolos para distribuição de rótulos como o LDP, RSVP-TE e o Roteamento por Segmentos, sem interação entre si. Cada nó de rede reserva uma faixa de rótulos dentro do SRGB para o plano de controle do Roteamento por Segmentos e rótulos fora do SRGB para alocação dinâmica pelo plano de controle MPLS tradicional. As entradas dos rótulos provenientes do LDP e do Roteamento por Segmentos são indexadas na FIB e LFIB dos roteadores. A Figura 5.38(a) mostra o sentido de um pacote da rede MPLS para a IP com rótulos

configurados pelo Roteamento por Segmentos (dentro da faixa do SRGB) e pelo LDP. As múltiplas entradas advindas do Roteamento por Segmentos ou pelo LDP podem ser programadas para o mesmo prefixo de destino conforme mostra a figura. Para o sentido do pacote da rede IP para a MPLS com Roteamento por Segmentos e LDP coexistindo da Figura 5.38(b), o rótulo a ser imposto ao pacote só pode ser recebido de um dos protocolos, não de forma simultânea. Se existem vários caminhos para o destino, a entrada da tabela deve definir se o caminho selecionado pelo protocolo de Roteamento por Segmentos ou pelo LDP deve ser usado. Por padrão, a escolha da entrada é a do protocolo LDP, sendo necessário ativar a preferência pelo Roteamento por Segmentos. Uma proposta de migração do LDP para Roteamento por Segmentos em uma rede de roteadores de núcleo IP/MPLS é inicialmente manter em execução os dois planos de controle independentes. Os roteadores de núcleo então são atualizados para Roteamento por Segmentos, e os roteadores são configurados para preferencialmente utilizar a imposição dos rótulos através do Roteamento por Segmentos. O passo final é remover o LDP dos roteadores, simplificando assim rede. O IETF vem trabalhando na elaboração de normas para padronizar a interoperabilidade de redes IP/MPLS com Roteamento por Segmentos e o protocolo LDP [Filsfils et al., 2015], permitindo a coexistência das redes legadas tradicionais IP/MPLS.



(a) Caso MPLS para IP.



(b) Caso IP para MPLS.

Figura 5.38. Coexistência entre o Roteamento por Segmentos e o LDP.

5.4.6. Servidor de Mapeamento

O Servidor de Mapeamento (*Mapping Server*) é usado na interoperabilidade entre nós habilitados e não-habilitados ao Roteamento por Segmentos. O mapeamento dos prefixos de rede e SIDs são configurados no Servidor de Mapeamento, de forma similar ao Refletor de Rotas (*Router Reflector* - RR). Esse mapeamento é realizado no servidor de mapeamento e é necessário para interoperabilidade das redes MPLS tradicionais que usam o protocolo LDP com as redes implementadas com Roteamento por Segmentos. O servidor de mapeamento é um mecanismo de controle, não necessitando estar localizado no plano de dados, e deve ser redundante. O cliente do mapeamento recebe e analisa os mapeamentos de prefixos para SIDs do servidor, que usa as entradas aprendidas e configuradas localmente nas tabelas RIB para construção de uma política válida e consistente de mapeamento de SIDs. A instância do IGP utiliza as políticas do mapeamento de SIDs para recalcular alguns ou todos prefixos SID. Uma regra de consenso é quando o servidor de mapeamento for usado, todos os nós devem se comportar como clientes para recebimento do mapeamento de prefixos de tal forma que não recebam os LSAs através de nós que não são habilitados com Roteamento por Segmentos. Os anúncios do servidor de mapeamento não são nem propagados entre áreas OSPF, nem tampouco entre áreas IS-IS. Os anúncios do servidor de mapeamento, tanto no IS-IS quanto no OSPF, são implementados nas extensões desses protocolos. É possível haver múltiplos servidores para anúncio de mapeamentos de prefixos para SIDs. Espera-se, porém, que o conjunto de mapeamentos seja igual para os servidores, a fim de se manter consistência. Caso não haja consistência, o cliente do servidor de mapeamento escolhe a entrada que não tenha sobreposição como política ativa. A arquitetura cliente-servidor de mapeamento é mostrada na Figura 4.16.

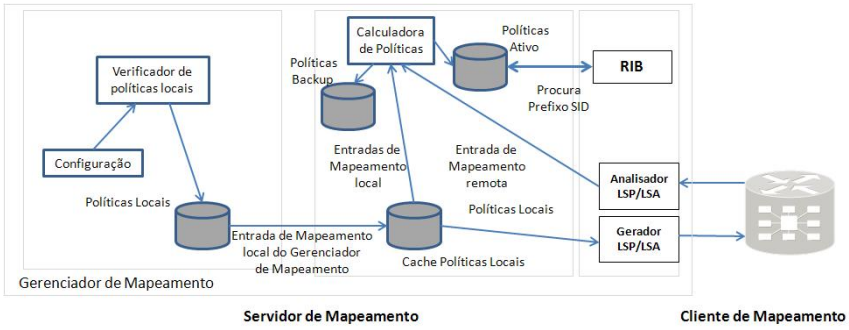
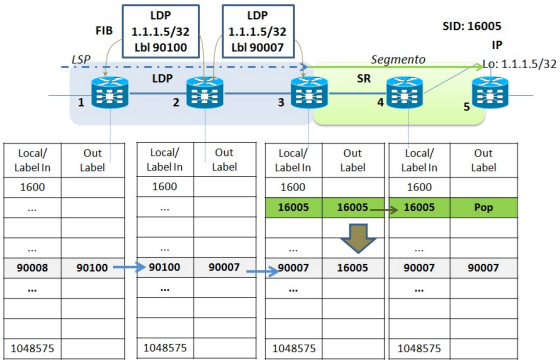


Figura 5.39. Arquitetura Cliente-Servidor de Mapeamento.

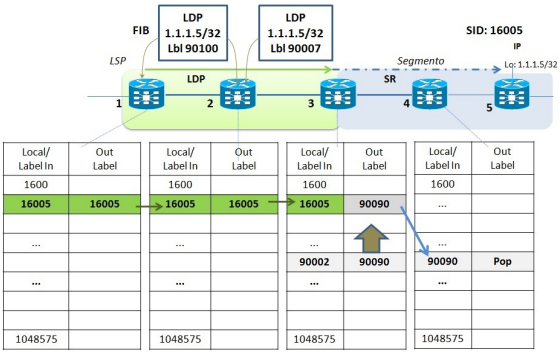
5.4.7. Interoperabilidade entre Roteamento por Segmentos e o protocolo LDP

Existem alguns modelos propostos para interoperabilidade do Roteamento por Segmentos com o protocolo LDP: Roteamento por Segmentos para LDP, LDP para Roteamento por Segmentos, Roteamento por Segmentos sobre LDP e LDP sobre Roteamento por Segmentos. O LDP para Roteamento por Segmentos conecta um LSP criado pelo LDP com um prefixo SID, em qualquer nó na borda do domínio LDP para Roteamento por Segmentos através de uma entrada na LFIB, configurada de forma automática em cada um dos domínios. O rótulo de entrada do nó de borda é obtido pelo LDP e o rótulo de saída é o do segmento, copiado do que seria o rótulo de entrada para o trecho

do segmento conforme mostrado na Figura 5.40(a). No Roteamento por Segmentos para LDP, como o destino é um domínio que não tem Roteamento por Segmentos habilitado (domínio LDP), é necessário anunciar no domínio LDP os prefixos SID na tabela LFIB destes roteadores. Isso é feito através do servidor de mapeamento, que informa o prefixo e a partir do prefixo IP é conhecido o rótulo para chegar ao destino conforme mostrado na Figura 5.40(a). Já no Roteamento de Segmentos sobre LDP, a cada borda de rede que utilize o Roteamento por Segmentos/LDP, o prefixo SID é mapeado em um LSP obtido do protocolo LDP. Na borda LDP/Roteamento por Segmentos o LSP é mapeado em um prefixo SID. Se o caminho terminar em um nó com domínio LDP, é necessário utilizar o servidor de mapeamento. Por fim, no LDP sobre Roteamento por Segmentos, na fronteira LDP/Roteamento por Segmentos, o LSP criado pelo LDP é mapeado em um prefixo SID, sendo portanto, necessário o servidor de mapeamento.



(a) LDP para Roteamento por Segmentos.



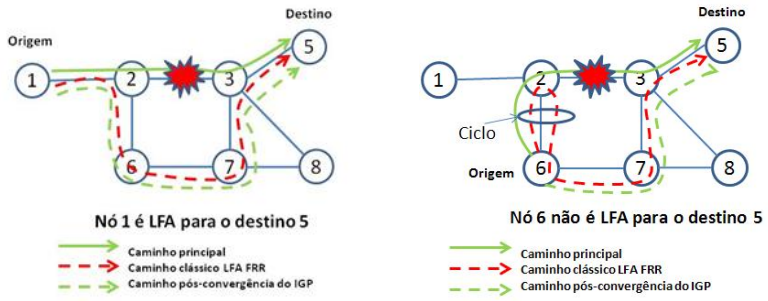
(b) Roteamento por Segmentos para LDP.

Figura 5.40. Interoperabilidade entre o Roteamento por Segmentos e o LDP.

5.4.8. Mecanismo de proteção - Topology Independent LFA (TI-LFA)

O mecanismo TI-LFA (*Topology Independent Loop Free Alternate*) é uma técnica de re-roteamento para convergência rápida [Francois et al., 2015]. O tempo de convergência deve ser em menos de 50ms, tempo inferior à convergência do IGP da rede. O LFA clássico em redes IP/MPLS depende da topologia, nem sempre provendo o melhor caminho de proteção. No mecanismo clássico LFA-FRR (*Loop Free Alternate Fast Reroute*), o IGP pré-calcula um caminho de proteção para cada caminho primário, instalando

o caminho de proteção no plano de dados. Na ocorrência de uma falha, todos os caminhos de proteção dos destinos impactados são habilitados com prefixos existentes na rede, convergindo em 50ms. O mecanismo clássico LFA-FRR possui algumas desvantagens como cobertura incompleta e caminho de proteção nem sempre ótimo. O LFA clássico é dependente da topologia e nem toda topologia é livre de ciclos para todos os destinos conforme mostra a Figura 5.41. Na Figura 5.41(a), o caminho principal sai do nó de origem 1 ao nó de destino 5. O caminho de proteção tem os nós intermediários 2, 6 e 7 livres de ciclos, coincidindo com o caminho que o IGP tomaria após a convergência da rede (maior que 50ms). Na Figura 5.41(b), um exemplo de caminho principal saindo do nó de origem 6 para o nó de destino 5. Nesse caso, o LFA clássico tem cobertura incompleta (nem todos os nós são LFA) e é dependente da topologia, nem sempre provendo um caminho ótimo de proteção, com a presença de ciclos. Uma topologia independente provê 100% de cobertura de nós com LFA e esse tipo de mecanismo de proteção é comumente usado em Roteamento por Segmentos.



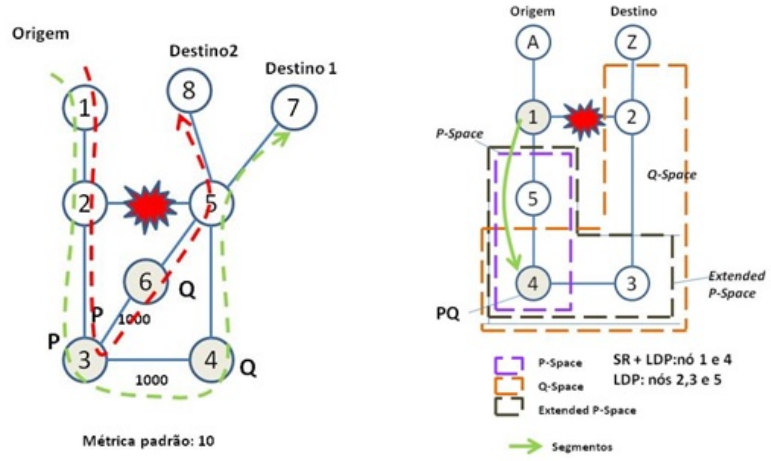
(a) Exemplo sem ciclo (LFA). (b) Exemplo com ciclo.

Figura 5.41. Mecanismo LFA clássico.

O TI-LFA usa o caminho pós-convergência com um caminho de proteção de reroteamento rápido. O mecanismo TI-LFA foi elaborado para ter 100% de cobertura dos nós sem ciclos e com tempo de convergência $\leq$50ms, prevenindo congestionamento transiente e roteamento não-ótimo. O caminho otimizado e natural após uma falha seria o caminho pós-convergência calculado pelo IGP, utilizado pelo TI-LFA. No entanto, não existe uma garantia de convergência rápida ($\leq$50ms), além dos nós serem livres de ciclos para este novo caminho. O Roteamento por Segmentos força o caminho de pós-convergência através de uma lista de segmentos. O mecanismo do TI-LFA influencia o mecanismo clássico de LFA, forçando a ausência de ciclos em um caminho ótimo, encontrado pelo IGP após o período de convergência da rede.

O mecanismo TI-LFA utiliza o algoritmo "PQ", que encontra nós que satisfaçam as propriedades de "P" e "Q". O algoritmo é proprietário Cisco e não está no escopo na padronização do IETF para o TI-LFA. O TI-LFA pode ser usado para balanceamento de tráfego, e proteção de tráfego MPLS com protocolos de controle tradicionais como o LDP. O TI-LFA utiliza caminhos ECMP (*Equal-Cost Multi-Path routing*), permitindo o balanceamento de tráfego baseado em uma função hash. A Figura 4.21a mostra um exemplo de balanceamento de tráfego do nó origem 1 para os nós de destino 7 e 8. O enlace a ser protegido é o enlace 2-5 (enlace do caminho principal ou primário). O TI

LFA calcula dois pares de nós “P” e “Q” para ambos os destinos 7 e 8, neste exemplo para o destino 10 (P=3, Q=4) e para o destino (P=3, Q=6). O TI LFA estatisticamente faz o balanceamento do tráfego nestes pares de nós. O tráfego MPLS controlado pelo LDP e não por Roteamento por Segmentos, também pode fazer uso do TI-LFA, eliminando sessões T-LDP (*Target LDP*), e o Roteamento por Segmentos pode ser implementado em “ilhas” dentro da rede controlada pelo LDP, influenciando a proteção do TI LFA para redes MPLS com LDP. A FIB (*Fowarding Information Base*) usa a fusão de rótulos obtidos do Servidor de Mapeamento (*Mapping Server*), que anuncia os prefixos para segmentos referentes ao caminho de proteção. Os nós de onde partem os caminhos de proteção devem ter Roteamento por Segmentos habilitados, bem como o nó de destino, que deve ser associado a um prefixo SID anunciado por este nó, ou através de um Servidor de Mapeamento (*Mapping Server*). Além da origem e do destino, os nós P e Q também devem estar habilitados para Roteamento por Segmentos. O exemplo da Figura 4.21b mostra o nó 1 e 4 com Roteamento por Segmentos e LDP habilitados. O nó de destino “Z”,e os nós 2, 3 e 5 são configurados somente com LDP. O Servidor de Mapeamento (*Mapping Server*) anuncia o prefixo SID 16006 para a interface “loopback” do nó Z, cujo IP é por exemplo 1.1.1.6/32. O nó 4 é um nó do tipo PQ para proteção do nó de destino “Z” no nó 1 do caminho principal ou primário 1-2. O nó 1 e 4 utilizam as funcionalidades de interoperabilidade do Roteamento por Segmentos e do protocolo LDP para dirigir os pacotes para o caminho de proteção encontrado pelo TI LFA.



(a) Balanceamento de tráfego de proteção.

(b) TI LFA em redes MPLS com LDP.

Figura 5.42. Exemplos de implementação de TI LFA com Roteamento por Segmentos.

5.4.9. Aplicações em Roteamento por Segmentos

A Engenharia de Tráfego é um exemplo de aplicação no contexto de Roteamento por Segmentos (*Segment Routing - Traffic Engineering* - SR-TE) utilizando os benefícios das Redes Definidas por Software e do Roteamento por Segmentos [Bhatia et al., 2015]. O Roteamento por Segmentos permite a configuração, a modificação e a remoção de caminhos TE dentro de um domínio de rede, operando somente na borda da rede. O plano

de controle do Roteamento por Segmentos pode ser mantido tanto de forma centralizada através de um controlador SDN, quanto de forma distribuída no plano de controle existente nos roteadores de núcleo habilitados com Roteamento por Segmentos. Neste último, os roteadores podem pertencer a uma rede composta somente por nós que executem o Roteamento por Segmentos ou em uma mista, que execute também o LDP. Na Engenharia de Tráfego com Roteamento por Segmentos, o roteador de núcleo de origem (*Head End LSR*) é denominado roteador de núcleo de origem SRTE (*Segment Routing Traffic Engineering Head End*), onde os pacotes são classificados e onde são impostas as listas de segmentos. Todas as funcionalidades de engenharia de tráfego são influenciadas pelas decisões do Roteamento por Segmentos e pelo controlador SDN. Os roteadores intermediários (*Mid Point*), para efeito do Roteamento por Segmentos não existem em sua topologia lógica, não necessitando de estados adicionais (troca de rótulos e atualizações de tabelas LFIBs) e sinalizações (protocolo RSVP-TE). A centralização da engenharia de tráfego permite melhorar a otimização, a previsibilidade e a convergência da rede através de aplicações programáveis por APIs NBI e de mecanismos de programação da rede como o PCEP e APIs SBI. O PCE e o BGP-LS possuem extensões para Roteamento por Segmentos. O BGP-LS é usado para anunciar o estado do enlace e a base de dados TE (*Traffic Engineering Database* - TED). A informação do estado do enlace é repassado em LSAs opacos, incluindo a informação dos estados do Roteamento por Segmentos distribuídos pelo IGP. O PCE (*Path Computation Element*) especifica uma lista de SIDs a partir de uma requisição de indicação de caminho proveniente da aplicação, e o PCC (*Path Computation Client*) encaminha o tráfego impondo a lista de segmentos nos pacotes conforme mostrado na Figura 5.43. Não existe sinalização RSVP-TE e os caminhos podem ser iniciados pelo PCE ou PCC.

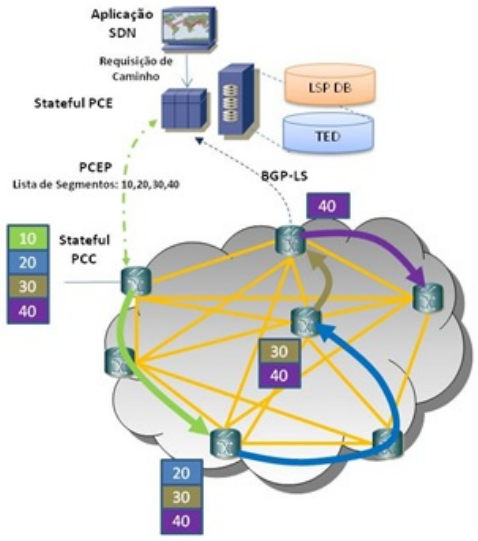


Figura 5.43. Roteamento por Segmentos com decisão de caminho por SDN.

A centralização da inteligência da engenharia de tráfego com uma visão fim a fim da rede pode ser obtida através da tecnologia de Roteamento por Segmentos e Redes Definidas por Sotware, onde controlador expressa o caminho em uma lista de segmentos, e a

rede mantém os segmentos e providencia o reroteamento rápido (*Fast Reroute* - FRR) para o mesmo, coincidente com os caminhos de menor custo (*Equal-cost multi-path routing* - ECMP).A Figura 5.44 mostra um controlador SDN fazendo a orquestração de engenharia de tráfego para descoberta de um caminho que atenda a demanda de banda de 2G para um túnel MPLS que vai de A a Z. Nas redes IP/MPLS legadas o RSVP-TE era responsável para obtenção dos recursos através de complexa sinalização. O controlador encontra o caminho a partir do dados coletados (através do BGP-LS por exemplo) usando estes dados na computação do caminho, que é uma lista de segmentos, configurados através uma "southbound interface" como NETCONF ou CLI conforme deduzido na figura.

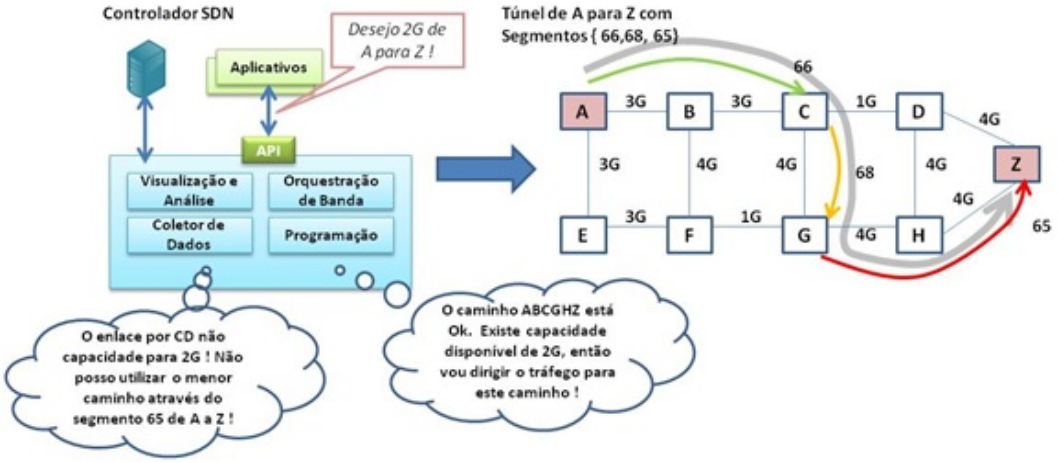


Figura 5.44. Aplicação SR TE com orquestração de banda.

Outra aplicação do SR TE é a determinação do caminho baseado em classe de serviço (*CoS-Based TE*). Nesse caso, o tráfego de dados pode ser direcionado por segmentos onde os enlaces possuem maior capacidade e menor custo por bit, enquanto o tráfego de voz sobre IP pode ser direcionado por segmentos com enlaces de menor latência. Para redes extensas e multi-áreas são usados prefixos SID "Anycast". Outra aplicação do SR TE para fins operacionais e de recuperação de rede é o controle OAM (*Operations, Administration and Maintainance*). O uso dos segmentos de adjacência permite monitorar cada enlace da rede, sendo bastante útil em redes extensas e complexas como a rede de roteadores de núcleo das operadoras de telecomunicações. A Figura 5.45 demonstra esse tipo de aplicação. O monitoramento do caminho ABCFGDH é realizado por segmentos de adjacência, permitindo localizar perda de pacotes em um enlace. Este tipo de aplicação é descrito na versão "draft" do IETF *draft-geib-spring-oam-usecas-02*.

O Roteamento por Segmentos automatiza o "peering" de roteadores de diferentes sistemas autônomos através da automação do BGP com alocação de SIDs, simplificando o "peering" da rede IP/MPLS legada. O controlador SDN aprende os SIDs dos "peers" e a topologia externa através do roteador de borda pelo BGP-LS EPE (*Egress Peering Engineering*), definidos nas extensões do BGP-LS. Esse tipo de caso de uso, definido no "draft" *draft-ietf-spring-segment-routing-central-epe*, é ilustrado na Figura 5.46.

Das aplicações com Roteamento por Segmentos com BGP, se destacam casos de

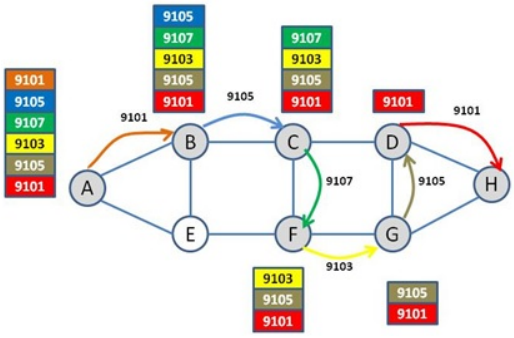


Figura 5.45. Aplicação OAM com roteamento de segmentos de adjacência.

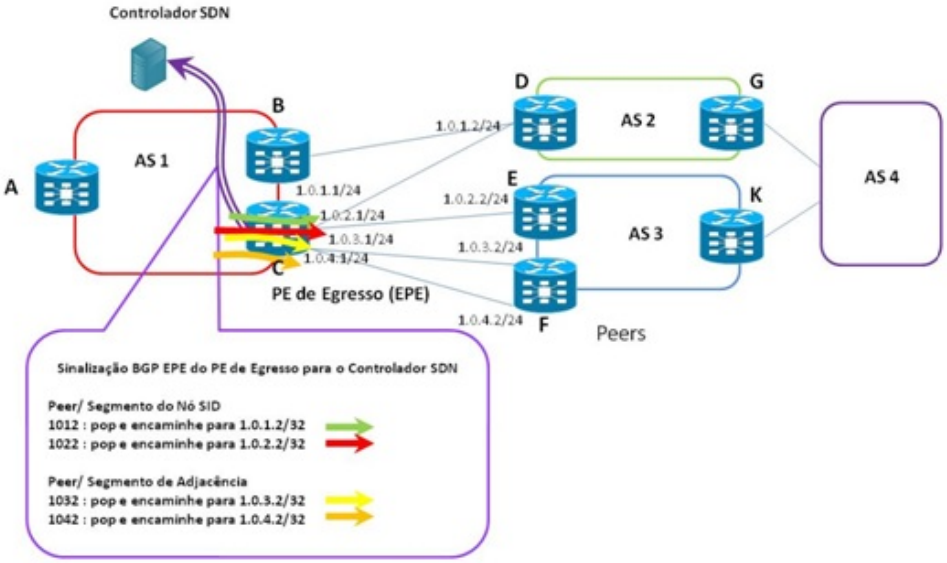


Figura 5.46. Aplicação EPE - automação de "peering" da rede.

uso para Datacenters em escala massiva (*Massive Scale DC* - MSDC) onde os prefixos de segmentos são usados no BGP assim como no IGP, definido no documento do IETF "draft" *draft-ietf-idr-bgp-prefix-sid*. Qualquer nó dentro da topologia aloca o mesmo segmento BGP para o mesmo comutador do Datacenter (*Top of Rack Switch* - TOR), com os benefícios do reroteamento rápido e engenharia de tráfego. Uma aplicação com plano de encaminhamento de dados IPv6 e também MPLS é a cadeia de serviços baseada em Roteamento por Segmentos. O nó que conecta a instância do serviço origina um SID baseado no comportamento do serviço. O nó pode ser virtual ou físico, os SIDs são conhecidos no nó de ingresso, através de um controlador SDN com múltiplas APIs como protocolos IGP e BGP, NETCONF, REST, OpenFlow, etc. Não existe sobrecarga da aplicação, nem manutenção de estados para cada cadeia, apenas um único estado por instância de serviço. Recentemente o IETF definiu uma proposta para transportar cadeias de serviço dentro de um cabeçalho NSH (*Network Service Header*) que identifica uma cadeia (*path id*) para transporte de meta-dados. O IETF e esté trabalhando para integrar o NSH com Roteamento por Segmentos como definido no "draft" Network Service Header *draft-ietf-*

sfc-nsh-04, que mapeia os segmentos dentro do *path id* como uma opção para redes de Datacenter. A Figura 5.47 mostra um exemplo de cadeia de serviços com Roteamento por Segmentos, podendo ser aplicável em NFV (*Network Functions Virtualization*). O Roteamento por Segmentos com plano de encaminhamento de dados IPv6 apresenta os mesmos estudos de caso do com plano de encaminhamento de dados MPLS.

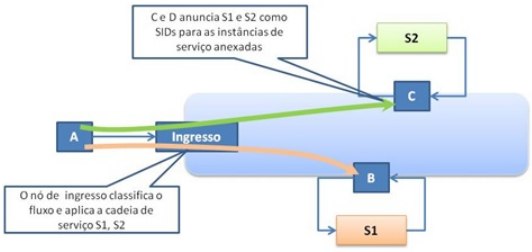


Figura 5.47. Cadeia de serviços com Roteamento por Segmentos.

5.4.10. Padronização IETF

A padronização da arquitetura de Roteamento por Segmentos está sendo conduzida pelo IETF, no grupo de trabalho SPRING WG (*Source Packet Routing in Networking Working Group*), cujas RFCs estão na versão "draft", que focam arquitetura, casos de uso e extensões dos protocolos IS-IS, OSPF, BGP, BGP-LS, PCEP e IPv6.

5.5. Experimentação Prática

Nos experimentos práticos deste minicurso, a ferramenta de simulação OSHI (*Open Source Hybrid IP/SDN networking*) [Davoli et al., 2015] é empregada. Essa ferramenta auxilia a investigação e prototipagem das redes Openflow, para que elas possam oferecer as mesmas funcionalidades das redes IP/MPLS. A ideia foi utilizar um simulador que permita trabalhar com nós de redes virtuais e reais, e com implementação de software aberto em IP e SDN. A interface *southbound* é Openflow e as interfaces *northbound* são APIs REST. O tráfego é enviado em vários tipos de túneis tais como MPLS, VLAN, Q-in-Q e Ethernet PBB (*Provider Backbone Bridge*) através do controlador SDN. Os sistemas operacionais de rede escolhidos foram o Floodlight, Ryu e ONOS [Salsano et al., 2014b, Stancu et al., 2015]. O simulador de rede híbrida IP/SDN possui três formas de implementação: no Virtual Box "run time", no ambiente de simulação do Mininet e em provas de conceito de redes SDN como o projeto OFELIA aplicando o conceito de experimento como serviço (*testbed as a service*) [Salsano et al., 2014b]. Nesta seção, o Mininet é usado [Salsano et al., 2014b, OSHI e Mininet, 2015]. O simulador utiliza nós virtuais de rede de núcleo denominados Roteadores de Núcleo em software aberto (*Open Source Label Switch Routers* - OpenLSR), que geram pacotes OSPF e LDP utilizando o Quagga e computam os rótulos MPLS que são instalados nos comutadores usando o protocolo OpenFlow. O controlador SDN também possui um módulo de software para engenharia de tráfego e roteamento por segmentos [Davoli et al., 2015].

5.5.1. Introdução ao Mininet

O Mininet [Mininet, 2015] é um emulador de rede que cria redes virtuais com servidores, comutadores, controladores e enlaces virtuais em uma única máquina física ou virtual. O Mininet possui linha de comando própria (*Command Line Interface* - CLI) e APIs que permitem a criação de rede e serviços, customização e compartilhamento com outros usuários. O Mininet é ideal para experimentos com OpenFlow e redes SDN e pode ser executado em um PC ou até mesmo em um laptop. Os passos seguintes ilustram a criação e o uso de uma topologia mínima, que inclui o controlador POX, um comutador OpenFlow e dois servidores.

Passo 1: Inicialização do Mininet com a topologia mínima.
Passo 2: Verificação dos nós da rede.
Passo 3: Verificação dos enlaces da rede.
Passo 4: Verificação dos endereços lógicos dos dispositivos da rede. Os nós são configurados em uma subrede 10.0.0.0/8 por padrão.
Passo 5: Acesso aos servidores HTTP através do terminal.
Passo 6: Teste da conectividade da rede através do `ping`.

O Mininet permite a criação de topologias mais complexas com mais comutadores e servidores. O Mininet ainda permite a construção de topologias através de APIs em Python, que é a base de construção do emulador. Os comandos passo-a-passo necessários para a execução deste primeiro experimento e de todos os outros a seguir estão disponíveis em `http://www.gta.ufrj.br/~silverio/SRArquivoSBRC2016.htm`.

5.5.2. Construção de uma rede simples com roteamento no Mininet

Este experimento visa criar uma rede simples com roteamento estático, onde o roteador recebe e processa pacotes como um roteador físico real, e depois os encaminha pelas interfaces corretas. O roteador emulado encaminha pacotes de um cliente para dois servidores HTTP.

Para tal, os seguintes passos devem ser seguidos:

Passo 1: Instalação do módulo do controlador POX.
Passo 2: Verificação dos arquivos de configuração.
Passo 4: Configuração do ambiente através da execução do arquivo `config.sh`.
Passo 5: Inicialização do controlador POX.
Passo 6: Realização dos testes propostos a partir da execução do arquivo `sr_solution`.
Passo 7: Inicialização dos testes propostos no Mininet.
Passo 8: Captura dos pacotes no formato `pcap` através do executável `sr`.

5.5.3. Open Source Hybrid IP/SDN Networking

A ferramenta OSHI (*Open Source Hybrid IP/SDN networking*) [OSHI, 2015] permite o uso de nós híbridos que combinam o encaminhamento tradicional IP com o encaminhamento SDN. O OSHI é uma ferramenta de código aberto que implementa comutadores OpenFlow (*OpenFlow Capable Switch* - OFCS), um encaminhador de pacotes IP e um *daemon* de roteamento IP. O OFCS é conectado ao conjunto de interfaces físicas pertencentes à rede IP/SDN, enquanto o encaminhador de pacotes IP é ligado a um conjunto de

portas virtuais OFCS, como visto na Figura 5.48. No nó OSHI, o OFCS é implementado usando o Open vSwitch, o encaminhador de pacotes IP é o kernel do Linux e o Quagga atua como o *daemon* de roteamento. .

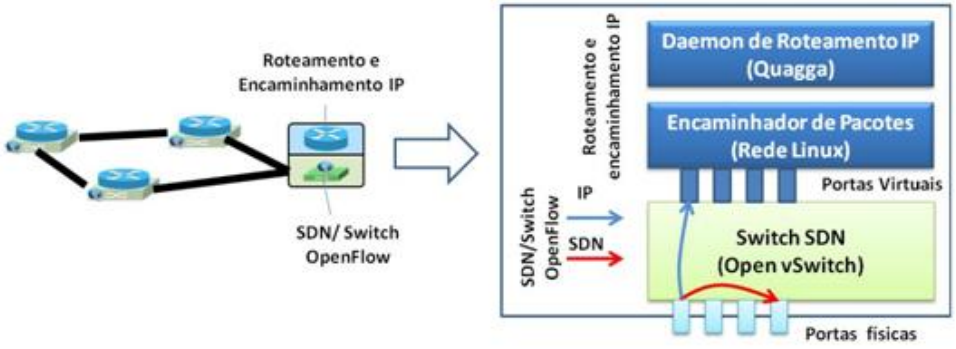


Figura 5.48. Construção de uma rede simples com roteamento no Mininet (Adaptado de [Salsano et al., 2014b]).

Os serviços a serem simulados pela ferramenta OSHI são do tipo IP ponto-a-ponto como Circuitos Virtuais Alugados (*Virtual Leased Lines* - VLL), Pseudo-fio (*Pseudowires* - PW) de camada 2 e comutadores virtuais de camada 2 (*Virtual Switched Services* -VSS) sobre um backbone IP/MPLS. Este último permite também simular engenharia de tráfego em redes IP/MPLS (túneis TE) e roteamento por segmentos. A fim de apoiar tanto o desenvolvimento, quanto os aspectos de teste e de avaliação comparativa, a ferramenta Mantoo (*Management Tools*) [Salsano et al., 2014b, Salsano et al., 2014a] foi adicionada para auxiliar experiências SDN sobre simuladores e redes experimentais distribuídas.O Mantoo inclui *scripts* Python para configuração e controle dos simuladores ou *scripts* para medições de desempenho e um visualizador de topologia 3D.A representação da topologia é através de arquivos no formato JSON e a interface gráfica possui um módulo de criação automático de topologia a partir dos dados desses arquivos. Os *scripts* de configuração incluem um analisador da topologia, extensões das bibliotecas do Mininet e o configurador OSHI.

5.5.4. Criação de um serviço MPLS através da ferramenta OSHI-TE

A ferramenta OSHI-TE utiliza nós virtuais de rede de núcleo chamados OpenLSR (*Open Source Label Switch Routers*), que geram pacotes OSPF e LDP utilizando o Quagga. Este último computa os rótulos MPLS que são instalados nos computadores usando o protocolo OpenFlow. O controlador SDN também possui um módulo de software para engenharia de tráfego e roteamento por segmentos [SPRING, 2015].

Os serviços simulados pelo OSHI são implementados como caminhos em uma rede SDN, a partir de controladores Ryu ou Floodlight centralizados. Duas propostas foram concebidas e implementadas para o estabelecimento de caminhos, a primeira baseia-se em identificadores de VLAN e a segunda em rótulos MPLS. Como exemplo, o serviço Pseudo-fio foi implementado usando apenas os rótulos MPLS, a partir do encapsulamento do pacote Ethernet cliente.O serviço de Pseudo-fio em uma rede híbrida IP/SDN tem as mesmas características dos serviços implementados em rede tradicional

IP/MPLS. O controlador utilizado foi o Ryu e um *script*, denominado `VLLPusher` que utiliza a API REST do controlador foi utilizado para recuperar a topologia de rede e, em seguida, avaliar o caminho mais curto entre os pontos finais do serviço. Nesse passo, é possível introduzir os aspectos de engenharia de tráfego [Davoli et al., 2015]. Finalmente, o *script* aloca os rótulos MPLS e usa a API REST do controlador Openflow para definir as regras para o encaminhamento de pacotes e a comutação de rótulos MPLS.

A arquitetura do nó OSHI permite não somente a criação dos serviços de Pseudo-fio, mas também de comutadores virtuais. O comutador OpenFlow suporta a inserção e retirada de rótulos, enquanto o ACE (*Access Encapsulator*) provê o túnel GRE. O ACE é implementado como uma nova instância do Open vSwitch, utilizando duas funções de virtualização: espaço de nomes de redes e pares de portas Ethernet virtuais.

A seguir são enumerados os passos para a criação de um serviço de Pseudo-fio, que pode ser executado através da linha de comando ou da interface gráfica Mantoo:

Passo 1: Execução do *script* de carga da topologia de um serviço Pseudo-fio através do Mininet ou da interface gráfica.
Passo 2: Realização de testes de conectividade e captura de pacotes com farejadores de rede.
Passo 3: Acesso remoto ao controlador e verificação das saídas dos *scripts*.
Passo 4: Teste da conectividade entre os roteadores clientes.
Passo 5: Configuração dos circuitos virtuais.
Passo 6: Criação dos circuitos virtuais.
Passo 7: Teste da conectividade (Passo 4) e verificação dos serviços operacionais.
Passo 8: Remoção dos circuitos virtuais.

5.5.5. Criação de um serviço em rede MPLS-TE com roteamento por segmentos

Considerando o exemplo de rede MPLS da Figura 5.49, o roteamento por segmentos é baseado na inserção, retirada e comutação de rótulos que representam os segmentos. Na figura, o nó C e F anunciam seus segmentos globais 69 e 90 com os endereços de suas interfaces loopback para os outros nós através do IGP. Já o nó E anuncia o rótulo 23 como segmento de adjacência. Dessa forma, se o nó A enviar um pacote para F, ele deve criar uma lista de segmentos contendo os segmentos {90, 23, 69}. Os segmentos AC e EF correspondem aos menores caminhos de A para C e de E para F e, por conseguinte, devem ser os caminhos encontrados pelo IGP para alcançar o destino.

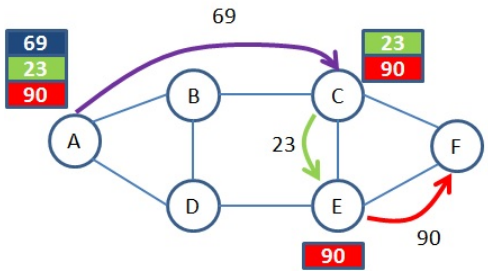


Figura 5.49. Caso de uso de roteamento por segmentos simplificado.

O projeto de rede com nós OSHI pode acomodar o roteamento por segmentos graças à característica híbrida IP/SDN da arquitetura do nó. Algumas premissas foram consideradas no ajuste da ferramenta para roteamento por segmentos: os segmentos utilizam rótulos MPLS, segmentos locais não são suportados, são utilizados os 16 bits mais significativos à direita correspondente à interface "loopback" para codificar o rótulo, cada interface OSHI em um nó tem o mesmo endereço físico e existe um mapeamento estático de MAC entre os nós OSHI usados para roteamento por segmentos. A ação de retirada do rótulo ocorre no último nó OSHI e não no penúltimo nó. Como na ferramenta não existe um plano de dados e de controle MPLS, este pode ser replicado utilizando tabelas OpenFlow e o uso de comutadores habilitadas com OpenFlow.

A seguir são enumerados os passos para a criação de um serviço VLL.

Neste exemplo, uma topologia gerada pela interface gráfica é analisada e um conjunto de fluxos é extraído para em seguida permitir a alocação de segmentos.

Passo 1: Verificação das extensões do roteamento por segmentos do Mininet.
Passo 2: Carregamento da topologia na interface gráfica.
Passo 3: Implantação da topologia.
Passo 4: Identificação do endereço IP do controlador e execução do *script* de implantação.
Passo 5: Geração de um catálogo de fluxos para ser manuseado pelo algoritmo de alocação de roteamento por segmentos.
Passo 6: Execução do algoritmo de roteamento por segmentos a partir da interface gráfica do OSHI.
Passo 7: Execução do projeto.
Passo 8: Execução do aplicativo `sr_vll_pusher`.
Passo 9: Realização dos testes de conectividade no VLL.
Passo 10: Remoção dos circuitos virtuais e término da emulação do Mininet.

5.5.6. Engenharia de Tráfego através do OSHI-TE

Para implementação de engenharia de tráfego a ferramenta OSHI utiliza um aplicativo que influencia as decisões do controlador Ryu através de uma API REST. Na entrada, é necessário um arquivo de configuração no formato JSON que descreve as relações do tráfego, enquanto que a topologia e capacidade dos enlaces são obtidas da API REST do módulo de topologia e do módulo do comutador habilitado com OpenFlow, que provê a topologia e velocidade das portas. A implementação do TE é dividida em três partes: a obtenção das entradas, algoritmos baseados em heurísticas e a instalação de regras. O último passo é alcançado pela API REST, que permite implantar as regras nos comutadores OpenFlow.

5.6. Considerações Finais e Direções Futuras

O roteamento por segmentos é uma proposta emergente para simplificação do roteamento e da configuração das redes das operadoras de telecomunicações. No Roteamento por Segmentos, os estados por fluxo são mantidos apenas nos nós de borda da rede e a configuração das redes de núcleo pode se tornar automatizada. Para tal, o Roteamento por Segmentos utiliza os benefícios da programação das redes definidas por

software através da centralização do plano de controle. A visão centralizada do controlador permite o cálculo dinâmico dos segmentos e a construção de rotas entre os nós de borda [Sgambelluri et al., 2015b].

O roteamento por segmentos tende futuramente a ser aplicado em redes de transporte ópticas com GMPLS, que utilizam instâncias hierárquicas de sessões de sinalizações. Tais sessões de sinalização devem ser estabelecidas também nos nós de trânsito das redes de transporte ópticas, tendo como consequência uma implementação complexa do plano de controle. O roteamento por segmentos tem grande potencial de integração entre as camadas de rede de transporte óptica e de roteadores de núcleo, permitindo a convergência IP e óptica [Sgambelluri et al., 2015a] e, consequentemente, proporcionando inúmeros benefícios para a operadora de telecomunicações. Dentre os benefícios estão o uso racional e otimizado da capacidade da rede, com proteção e restauração de acordo com de níveis de serviço diferenciados por fluxo de dados. Outras direções futuras indicam a aplicação do conceito de roteamento por segmentos em redes Carrier Ethernet [Cai et al., 2014, Bidkar et al., 2014], através da adição de rótulos com os segmentos no quadro Ethernet ("*Ominipresent Ethernet*"). Outras propostas de implementação do roteamento por segmentos em redes Carrier Ethernet são através do MPLS-TP (*MPLS Transport Profile*), que permite um uso misto em redes Carrier Ethernet baseadas em comutadores metro Ethernet e redes ópticas de transporte ou através do uso do encapsulamento do Ethernet com MPLS. O roteamento por segmentos ainda possui pontos em desenvolvimento, como questões de segurança nas extensões do cabeçalho IPv6; bem como integração com outras tecnologias, como por exemplo em virtualização de funções de redes. Nessa última, o roteamento por segmentos é usado para transportar a informação da cadeia de serviços, interoperando com o cabeçalho de serviços de rede, que ainda está em curso de padronização.

Referências

[Adrian et al., 2015] Adrian, G., Vasileios, K. e Xenofontas, D. (2015). Evaluating the effect of centralization on routing convergence on a hybrid bgp-sdn emulation framework. *Computer communication review*, 44:369–370.

[Alvarez, 2016] Alvarez, S. (2016). BRKMPL-2100 - deploying MPLS traffic engineering. Acessado em fev/2016 `https://www.ciscolive.com/online/connect/sessionDetail.ww?SESSION_ID=7818&backBtn=true`.

[Andreas et al., 2015] Andreas, B., Arsany, B., Martin, R. e Wolfgang, K. (2015). Survey on network virtualization hypervisors for software defined networking. *IEEE Communications Surveys and Tutorials*, 18:655–685.

[Asati, 2012] Asati, R. (2012). BRKIPM-2017 - designing IP/MPLS VPN networks. Acessado em fev/2016 `https://www.ciscolive.com/online/connect/sessionDetail.ww?SESSION_ID=2636&backBtn=true`.

[Bhatia et al., 2015] Bhatia, R., Hao, F., Kodialam, M. e Lakshman, T. (2015). Optimized network traffic engineering using segment routing. Em *IEEE Conference on Computer Communications (INFOCOM)*, p. 657–665.

[Bidkar et al., 2014] Bidkar, S., Gumaste, A., Ghodasara, P., Hote, S., Kushwaha, A., Patil, G., Sonnis, S., Ambasta, R., , Nayak, B. e Agrawal, P. (2014). Field trial of a software defined network (SDN) using carrier ethernet and segment routing in a tier-1 provider. Em *IEEE Global Communications Conference (GLOBECOM)*, p. 2166–2172.

[Bierman et al., 2015] Bierman, A., Bjorklund, M. e Watsen, K. (2015). RESTCONF Protocol. https://tools.ietf.org/html/draft-ietf-netconf-restconf-09.

[Cai et al., 2014] Cai, D., Wielosz, A. e Wei, S. (2014). Evolve Carrier Ethernet architecture with SDN and segment routing. Em *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, p. 1–6.

[Casado et al., 2014] Casado, M., Foster, N. e Guha, A. (2014). Abstractions for software defined networks. *Communications of the ACM*, 57:86–95.

[Casellas et al., 2015] Casellas, R., Munoz, R., Martinez, R., Vilalta, R., Liu, L., Tsuritani, T., Morita, I., Lopez, V., de Dios, O. G. e Fernandez-Palacios, J. P. (2015). SDN orchestration of openflow and GMPLS flexi-grid networks with a stateful hierarchical PCE. *IEEE Communications Magazine*, 7:106–117.

[Civanlar et al., 2015] Civanlar, S., Lokman, E., Kaytaz, B. e Tekalp, A. M. (2015). Distributed management of service-enabled flow-paths across multiple sdn domains. *IEEE Networks and Communications (EuCNC) 2015 European Conference on*, p. 360–364.

[Davoli et al., 2015] Davoli, L., Veltri, L., Ventre, P. L., Siracusano, G. e Salsano, S. (2015). Traffic engineering with segment routing: SDN-based architectural design and open source implementation. Em *European Workshop on Software Defined Networks (EWSDN)*, p. 111–112.

[De Ghein, 2007] De Ghein, L. (2007). *MPLS Fundamentals - A Comprehensive Introduction to MPLS Theory and Practice*. Cisco Press, 1 edição.

[development team, 2016] development team, R. (2016). Ryu Documentation Release 4.1. https://media.readthedocs.org/pdf/ryu/latest/ryu.pdf.

[El-Sayed e Jaffe, 2002] El-Sayed, M. e Jaffe, J. (2002). A view of telecommunications network evolution. *IEEE Communications Magazine*, 40:74–81.

[Farrel, 2006] Farrel, A. (2006). Introduction to the Path Computation Element. https://www.itu.int/ITU-/worksem/ngn/200604/presentation/s4_farrell.pdf.

[Feamster et al., 2004] Feamster, N., Balakrishnan, H., Rexford, J., Shaikh, A. e van der Merwe, J. (2004). The case for separating routing from routers. *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, p. 5–12.

[Filsfils et al., 2015] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Telecom, B., Ytti, S., Henderickx, W., Tantsura, J. e Crabbe, E. (2015). Segment Routing interoperability with LDP. https://www.ietf.org/archive/id/draft-filsfils-spring-segment-routing-ldp-interop-03.txt.

[Francois et al., 2015] Francois, P., Filsfils, C., Bashandy, A., Decraene, B. e Litkowski, S. (2015). Topology Independent Fast Reroute using Segment Routing. https://www.ietf.org/archive/id/draft-francois-spring-segment-routing-ti-lfa-02.txt.

[Gude et al., 2016] Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N. e Shenker, S. (2016). NOX: Towards an operating system for networks. http://www.cs.yale.edu/homes/jf/nox.pdf.

[Haleplidis et al., 2015] Haleplidis, E., Hadi Salim, J., Denazis, S. e Koufopavlou, O. (2015). Towards a network abstraction model for SDN. *Computer Communications*, 32:309–327.

[Hodzic e Zoric, 2008] Hodzic, H. e Zoric, S. (2008). Traffic engineering with constraint based routing in MPLS networks. Em *International Symposium ELMAR*, p. 269–272.

[Hu et al., 2014] Hu, F., Hao, Q. e Bao, K. (2014). A survey on software-defined network and openflow: From concept to implementation. *IEEE Communications Surveys & Tutorials*, 16:2181 –2206.

[Huang et al., 2014] Huang, S., Griffioen, J. e Calvert, K. L. (2014). Network hypervisors: Enhancing sdn infrastructure. *Computer Communications*, 46:87–96.

[IETF MPLS documents, 2001] IETF MPLS documents (2001). MPLS IETF WG. Acessado em fev/2016 `https://datatracker.ietf.org/wg/mpls/documents/`.

[ISO/IEC 23271:2012, 2012] ISO/IEC 23271:2012 (2012). Information technology – Common Language Infrastructure (CLI). Acessado em mar/2016 `http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=58046`.

[J., 2011] J., K. (2011). The GMPLS controlled optical networks as industry communication platform. *IEEE Transactions on Industrial Informatics*, 7:671–678.

[Je e Ly, 2012] Je, B. e Ly, O. (2012). Next-generation optical network architecture and multidomain issues. *Proceedings of the IEEE*, 100:1130–1139.

[Jingjing et al., 2014] Jingjing, Z., Di, C., Weiming, W., Rong, J. e Xiaochun, W. (2014). The deployment of routing protocols in distributed control plane of sdn. *The Scientific World Journal*, 44:1–8.

[Kaur et al., 2016] Kaur, S., Singh, J., e Ghumman, N. S. (2016). Network programmability using POX controller. http://www.sbsstc.ac.in/icccs2014/Papers/Paper28.pdf.

[Kreutz et al., 2015] Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S. e Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103:14–76.

[Lara et al., 2014] Lara, A., Kolasani, A. e Ramamurthy, B. (2014). Network innovation using openflow: A survey. *IEEE Communications Survey & Tutorials*, 16:493–512.

[Marzo et al., 2003] Marzo, J. L., Calle, E., Scoglio, C. e Anjah, T. (2003). QoS online routing and MPLS multilevel protection: a survey. *IEEE Communications Magazine*, 41:126–132.

[Matias et al., 2015] Matias, J., Garay, J., Toledo, N., Unzilla, J. e Jacob, E. (2015). Toward an SDN-enabled NFV architecture. *IEEE Communications Magazine*, 53:187–193.

[Medved et al., 2014] Medved, J., Varga, R., Tkacik, A. e Gray, K. (2014). Opendaylight: Towards a model-driven SDN controller architecture. volume 46, p. 1–6.

[Mininet, 2015] Mininet (2015). Mininet tutorial. Acessado em `https://github.com/mininet/mininet/wiki/Teaching-and-Learning-with-Mininet`.

[Nunes et al., 2014] Nunes, B., Antipolis, S., Mendonca, M., Nguyen, X. N. e Obraczka, K. (2014). A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials*, 16:1617–1634.

[Nwa et al., 2010] Nwa, S., Ju, L., Martin, R. e Phil, S. (2010). Automating network and service configuration using netconf and yang. *IEEE Communications Magazine*, 48:166–173.

[Osborne e Simha, 2002] Osborne, E. e Simha, A. (2002). *Traffic Engineering with MPLS*. Cisco Press, 2 edição.

[OSHI, 2015] OSHI (2015). OSHI - open source hybrid IP/SDN tutorial. Acessado em `http://netgroup.uniroma2.it/OSHI`.

[OSHI e Mininet, 2015] OSHI e Mininet (2015). OSHI - open source hybrid IP/SDN networking and its emulation on Mininet and on distributed SDN testbeds. Acessado em `htttp://netgroup.uniroma2.it/twiki/bin/view/Oshi/WebHome#AnchorSegRouting`.

[Paolucci et al., 2013] Paolucci, F., Cugini, F., Giorgetti, A., Sambo, N. e Castoldi, P. (2013). A survey on the path computation element (PCE) architecture. *IEEE Communications Surveys & Tutorials*, 15:1819–1841.

[Pepelnjak e Guichard, 2003] Pepelnjak, I. e Guichard, J. (2003). MPLS and VPN architectures. volume 1. Cisco Press.

[Pingping et al., 2015] Pingping, L., Jun, B., Stephen, W., Yangyang, W., Anmin, X., Ze, C., Hongyu, H., Yikai, L. e Pingping, L. (2015). A west-east bridge based sdn inter-domain testbed. *IEEE Communications Magazine*, 53:190–197.

[Previdi et al., 2015a] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B. e Tantsura, J. (2015a). IS-IS Extensions for Segment Routing. https://tools.ietf.org/html/draft-ietf-isis-segment-routing-extensions-06.

[Previdi et al., 2015b] Previdi, S., Filsfils, C., Field, B., Leung, I., Linkova, J., Aries, E., an E. Vyncke, T. K. e Lebrun, D. (2015b). IPv6 Segment Routing Header (SRH). https://www.ietf.org/archive/id/draft-previdi-6man-segment-routing-header-08.txt.

[RFC1157, 1990] RFC1157 (1990). A Simple Network Management Protocol (SNMP). Acessado em mar/2016 `http://www.ietf.org/rfc/rfc1157.txt?number=1157`.

[RFC5440, 2009] RFC5440 (2009). Path Computation Element (PCE) Communication Protocol (PCEP). Acessado em mar/2016 `https://tools.ietf.org/html/rfc5440`.

[RFC6020, 2010] RFC6020 (2010). YANG - a data modeling language for the network configuration protocol (NETCONF). Acessado em mar/2016 `https://tools.ietf.org/html/rfc6020`.

[RFC6241, 2011] RFC6241 (2011). Network Configuration Protocol (NETCONF). Acessado em mar/2016 `https://tools.ietf.org/html/rfc6241`.

[RFC7047, 2013] RFC7047 (2013). The Open vSwitch Database Management Protocol. Acessado em mar/2016 `https://tools.ietf.org/html/rfc7047`.

[RFC7752, 2016a] RFC7752 (2016a). North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP). Acessado em mar/2016 `https://www.rfc-editor.org/rfc/pdfrfc/rfc7752.txt.pdf`.

[RFC7752, 2016b] RFC7752 (2016b). North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP. https://datatracker.ietf.org/doc/rfc7752/.

[Rose, 2014] Rose, E. (2014). BRKMPL-1101 - understanding MPLS. Acessado em fev/2016 `https://www.ciscolive.com/online/connect/sessionDetail.ww?SESSION_ID=77795&backBtn=true`.

[Rothenberg et al., 2011] Rothenberg, C. E., Nascimento, M. R., Salvador, M. R. e Magalhães, M. F. (2011). Openflow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes. *Cad. CPqD Tecnologia*, 7:65–76.

[Rowshanrad et al., 2014] Rowshanrad, S., Namvarasl, S., Abdi, V., Hajizadeh, M. e Keshtgary, M. (2014). A survey on SDN, the future of networking. *Journal of Advanced Computer Science & Technology*, 3:232–248.

[Sadok e Kamienski, 2000] Sadok, D. e Kamienski, C. A. (2000). Qualidade de serviço na internet. Em *Minicursos do XVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, p. 4–44.

[Salsano et al., 2014a] Salsano, S., Pier, Ventre, L., Prete, L., Siracusano, G., Gerola, M., Salvadori, E., Santuari, M., Mauro, Campanella e Prete, L. (2014a). OSHI - open source hybrid IP/SDN networking and Mantoo - management tools for SDN experiments. Em *European Workshop on Software Defined Networks (EWSDN)*, p. 123–124.

[Salsano et al., 2014b] Salsano, S., Ventre, P. L., Prete, L., Siracusano, G., Gerola, M. e Salvadori, E. (2014b). OSHI - open source hybrid IP/SDN networking (and its emulation on mininet and on distributed SDN testbeds). Em *European Workshop on Software Defined Networks (EWSDN)*, p. 13–18.

[Santos et al., 2007] Santos, C. B., Fernandes, D. C. e Marchetti, B. R. B. (2007). Tutoriais TELECO MPLS: Re-roteamento dinâmico em redes IP utilizando network simulator. Acessado em fev/2016 `http://www.teleco.com.br/tutoriais/tutorialmplsrd/default.asp`.

[Scott-Hayward, 2015] Scott-Hayward, S. (2015). Design and deployment of secure, robust, and resilient sdn controllers. Em *IEEE Network Softwarization (NetSoft), 2015 1st IEEE Conference on*, p. 1–5.

[Sgambelluri et al., 2015a] Sgambelluri, A., Giorgetti1, A., Cugini, F., Bruno, G., Lazzeri, F. e Castoldi, P. (2015a). First demonstration of SDN-based segment routing in multi-layer networks. Em *Optical Fiber Communications Conference and Exhibition (OFC)*, p. 1–3.

[Sgambelluri et al., 2015b] Sgambelluri, A., Paolucci, F., Giorgetti, A., Cugini, F. e Castoldi, P. (2015b). Experimental demonstration of segment routing. *Journal of Lightwave Technology*, PP:1–1.

[SPRING, 2015] SPRING (2015). Source packet routing in networking (SPRING). Acessado em `https://datatracker.ietf.org/wg/spring/documents/`.

[Stancu et al., 2015] Stancu, A. L., Halunga, S., Vulpe, A., Suciu, G., Fratu, O. e Popovici, E. C. (2015). A comparison between several software defined networking controllers. *Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS), 2015 12th International Conference on*, p. 223–226.

[Systems, 2001] Systems, C. (2001). Implementing MPLS traffic engineering. Acessado em fev/2016 `http://www.cisco.com/c/en/us/td/docs/routers/crs/software/crs_r3-/mpls/configuration/guide/gc39crs1book_chapter4.html`.

[Telcordia GR-831, 1996] Telcordia GR-831 (1996). Operations Application Messages - Language For Operations Application Messages. Acessado em mar/2016 `http://telecom-info.telcordia.com/site-cgi/ido/docs.cgi?ID=SEARCH&DOCUMENT=GR-831&`.

[Xia et al., 2015] Xia, W., Wen, Y., Foh, C., Niyto, D. e Xie, H. (2015). A survey on software-defined networking. *IEEE Communications Surveys & Tutorials*, 17:27–51.

[Xiao et al., 2000] Xiao, X., Hannan, A., Bailey, B. e Ni, L. M. (2000). Traffic engineering with MPLS in the internet. *IEEE Network*, 14:28–33.

[Xiea et al., 2015] Xiea, J., Guo, D., Hua, Z., Qua, T. e Lvb, P. (2015). Control plane of software defined networks: A survey. *Computer communications*, 67:1–10.

Capítulo

6

Computação em Névoa: Conceitos, Aplicações e Desafios

Antônio Augusto Teixeira Ribeiro Coutinho[1], Elisângela Oliveira Carneiro[1] e Fabíola Gonçalves Pereira Greve[2]

[1]*Departamento de Tecnologia (DTEC), Universidade Estadual de Feira de Santana (UEFS)*
[2]*Departamento de Ciência da Computação (DCC), Universidade Federal da Bahia (UFBA)*

Abstract

Currently, cloud-based solutions have become the predominant approach for the Internet of Things (IoT). However, to meet requirements such as mobility support, location awareness and low latency, current IoT proposals are fomenting a major shift from a centralized model towards a decentralized model. Seen in these terms, Fog Computing is a paradigm that extends Cloud Computing services to the edge of the network on a widely distributed level. This chapter introduces the concepts related to this area highlighting the differences between IoT, Cloud and Fog Computing. Technological aspects about the integration of these paradigms are addressed by examining existing solutions. Future research is discussed focussing on challenges involved in designing fog computing systems.

Resumo

Atualmente, soluções baseadas em Nuvem têm sido uma abordagem predominante para a Internet das Coisas (Internet of Things, IoT). Porém, visando atender requisitos como mobilidade, localidade e baixa latência as propostas correntes sobre IoT estão fomentando uma importante mudança de um modelo centralizado para um modelo descentralizado. Nesse sentido, a Computação em Névoa (Fog Computing) é um paradigma que estende os serviços de Nuvem para a borda da rede numa escala amplamente distribuída. Este capítulo introduz os conceitos relacionados com esta área destacando as diferenças entre IoT, Computação em Nuvem e Névoa. Aspectos tecnológicos sobre a integração destes paradigmas são abordados pelo exame de soluções existentes. Pesquisas futuras são discutidas pela análise de desafios envolvidos no projeto de sistemas em Névoa.

6.1. Introdução

Nos próximos anos, é esperado o surgimento de bilhões de novos objetos com capacidade de coletar, trocar informações e interagir com o ambiente de maneira inteligente. Contudo, a integração desses elementos para beneficiar simultaneamente diferentes setores da sociedade demanda grandes desafios, como parte do que está sendo chamada a Internet das Coisas (*Internet of Things*, IoT) [Atzori et al. 2010] [Gubbi et al. 2013].

Os ambientes IoT são caracterizados por diferentes tipos de conexão entre dispositivos heterogêneos, dispostos de forma local ou amplamente distribuídos, com capacidades de comunicação, processamento e armazenamento limitadas [Sehgal et al. 2012]. Esses componentes físicos embarcados com sensores e atuadores podem ser interconectados com outros recursos físicos ou virtuais [Nitti et al. 2015] para controle, processamento e análise de dados. Sua implementação envolve diferentes questões como confiabilidade, performance, segurança e privacidade [Atzori et al. 2010] [Gubbi et al. 2013].

Propostas sobre a Internet das Coisas vêm agregando diferentes tecnologias para possibilitar a composição de cenários de computação ubíqua e pervasiva baseados em elementos auto-configuráveis e inteligentes sobre uma infraestrutura de rede dinâmica e global [Botta et al. 2016]. Neste contexto, a Computação em Nuvem (Cloud Computing) [Mell and Grance 2010] se distingue como uma tecnologia madura e confiável, oferecendo capacidades virtualmente ilimitadas de processamento e armazenamento para suplantar as limitações dos dispositivos IoT envolvidos [Moura and Hutchison 2016].

Diferentes soluções centradas em Nuvem para Internet das Coisas são citadas como Nuvem das Coisas (*Cloud of Things*) em [Aazam et al. 2014] e Nuvem IoT (*Cloud IoT*) em [Botta et al. 2016]. Essas plataformas agregam vantagens da Computação em Nuvem [Khalid et al. 2016] para suporte ao crescente volume de dados produzidos pelos dispositivos IoT [ABI Research 2015]. Elas também oferecem o compartilhamento dinâmico de recursos entre aplicações verticais distintas e permitem o gerenciamento de sua infra-estrutura em Nuvem IoT através da Internet [Moura and Hutchison 2016].

Porém, a transição para Internet das Coisas não pode ser considerada uma simples aplicação da Computação em Nuvem. Seu estudo envolve um conjunto de novas questões e desafios, requerendo grandes esforços de investigação [Aazam et al. 2014] [Botta et al. 2016] [Díaz et al. 2016]. É necessário otimizar e implantar o conceito de Nuvem para prover conteúdo aos usuários através de uma plataforma IoT densa e geograficamente distribuída. As soluções em Nuvem para Internet das Coisas terão que suportar o modelo *Everything-as-a-Service* (XaaS) [Duan et al. 2015], onde o usuário pode acessar os dados a partir de qualquer dispositivo, em qualquer lugar, a qualquer momento.

Recentemente, a Computação em Névoa (*Fog Computing*) [Bonomi et al. 2012] vem atraindo interesse pelo seu potencial de satisfazer requisitos que não são atendidos por um modelo centralizado em Nuvem [Khalid et al. 2016]. Este paradigma estende os recursos computacionais disponíveis na Nuvem para a borda da rede visando apoio às soluções em IoT. Dessa forma, possibilita a execução de aplicativos em bilhões de objetos conectados para fornecer dados, processamento, armazenamento e serviços aos usuários.

Sua arquitetura introduz o suporte à análise de dados em tempo real, distribuindo o processamento analítico através dos recursos na Névoa [Bonomi et al. 2014]. Esta

abordagem reduz significativamente a quantidade de informações transferidas para a infraestrutura em Nuvem por capturar e processar os dados necessários diretamente em cada dispositivo na borda da rede. Além disso, permite que os dados transmitidos sejam mais significativos e acionáveis pela filtragem de informações em diferentes níveis de sua organização hierárquica [Bonomi et al. 2012] [Bonomi et al. 2014].

Esse movimento seletivo dos recursos computacionais, controle e tomada de decisões para as extremidades da rede é uma área emergente da engenharia de sistemas e ciência da computação. Seu estudo está no centro de vários domínios de aplicação da Internet das Coisas onde pesquisas vem buscando identificar requisitos, experimentar algoritmos e avaliar arquiteturas para esclarecer problemas relativos à sua implementação.

No entanto, prover conectividade e prestação de serviços em larga escala no cenário da Internet das Coisas não é uma tarefa simples [Yi et al. 2015b]. Devido a sua localização e organização, as redes de nevoeiro possuem natureza heterogênea e uma diversidade de conexões envolvidas [Luan et al. 2016]. Diferentes problemas de conectividade, confiabilidade, capacidade e atraso podem influenciar na disponibilidade e qualidade dos serviços oferecidos [Madsen et al. 2013] [Yi et al. 2015b].

A segurança também é um fator chave, uma vez que um número crescente de elementos passam a integrar e atuar diretamente na rede. Isso aumenta a probabilidade de falhas não identificadas, infecções nos sistemas, vulnerabilidades nos canais e riscos de invasão. Neste aspecto, a Computação em Névoa enfrenta novos desafios de segurança e privacidade que vão além daqueles herdados da Computação em Nuvem [Yi et al. 2015c].

Devido a indefinição de padrões para integração das tecnologias relacionadas com a Nuvem IoT, existe atualmente a falta de uma interface unificada e de um modelo de programação que ajude tanto a implementação de novas soluções quanto a portabilidade de aplicativos para a plataforma em Névoa [Yi et al. 2015b]. Embora diferentes iniciativas de padronização tenham surgido nos últimos anos [ETSI ISG MEC 2015] [OpenFog 2016], ainda é preciso um grande esforço para atender às necessidades de seus desenvolvedores.

O objetivo geral deste capítulo é abordar o estado da arte da Computação em Névoa, oferecendo uma visão dos fundamentos, das tecnologias e dos desafios envolvidos na área. Neste sentido, seu conteúdo pretende cobrir os seguintes pontos: (*i*) apresentar aplicações e métodos usados para estender processamento, armazenamento e aplicações em direção a borda da rede visando apoio às soluções em IoT; (*ii*) descrever como as soluções em Névoa podem tirar vantagem da sua proximidade às fontes de dados para atender requisitos como mobilidade, localidade e baixa latência; (*iii*) identificar os desafios associados com a mudança de paradigma promovido pela computação em Névoa.

O restante do capítulo está organizado como segue. Na Seção 6.2 são apresentados conceitos e características da Computação em Névoa e sua integração com outras tecnologias. Na Seção 6.3 é apresentado um estudo sobre aplicações e plataformas de Computação em Névoa, com ênfase em aspectos que definem suas arquiteturas. Na Seção 6.4 são discutidos desafios associados com a mudança promovida por este paradigma e as pesquisas futuras nesta área.

6.2. Fundamentos sobre Computação em Névoa

Neste tópico, é oferecida uma visão geral da Computação em Névoa abordando como as propostas atuais sobre a Internet das Coisas estão fomentando uma importante mudança de paradigma de um modelo centralizado para um modelo descentralizado. Um conjunto de tecnologias emergentes relacionadas são descritas a fim de identificar o seu escopo, requisitos e integração. Serão enfatizadas a maturidade dessas tecnologias, os problemas não resolvidos pela computação em Nuvem e as motivações para a computação em Névoa.

6.2.1. Internet das Coisas

Em 1999, antes de *Kevin Ashton* criar o termo Internet das Coisas enquanto trabalhava no Auto-ID Labs [Auto-ID 2016], seu significado já fazia parte do imaginário de autores de ficção científica em livros, filmes, seriados e desenhos animados. Essa tecnologia é descrita em [Atzori et al. 2010] e [Gubbi et al. 2013] como um mundo repleto de objetos inteligentes e conectados, que participam do cotidiano das pessoas muitas vezes sem serem percebidos. Neste cenário, são capazes de interagir com o ambiente, trocar informações, monitorar processos, coletar estados, analisar dados, obedecer comandos e executar ações de forma coordenada e proativa para atender as finalidades de seus usuários.

Dentre os mais recentes paradigmas da computação, a Internet das Coisas é apontada como uma tecnologia emergente em [Gartner Inc. 2015]. Este documento eletrônico apresenta um relatório gráfico atualizado todos os anos com expectativas na área de Tecnologia da Informação (TI) e Marketing Digital. O *Hype Cycle* [Gartner Inc. 2016] serve como referência mundial para governos e organizações sobre a relevância e amadurecimento das tecnologias, dividido em cinco fases: inovação (*Innovation Triger*), ápice (*Peak of Inflated Expectations*), desilusão (*Trough of Desillusion*), esclarecimento (*Slope of Enlightenment*) e plenitude (*Plateau of Productivity*).

A fase de inovação é um período inicial onde são estabelecidas provas de conceito e protótipos sobre o paradigma. Na fase de ápice acontece uma diminuição de suas expectativas com o lançamento de seus primeiros produtos ou aplicações. A fase de desilusão envolve adaptações que resultam em novas versões ou mesmo em sua obsolescência prematura. Caso ultrapasse esse período, novas aplicações podem ser descobertas na fase de esclarecimento para atingir seu potencial produtivo na fase de plenitude. A Figura 6.1 mostra que em 2015 a Internet das Coisas atingiu seu ápice de expectativas apoiada pelo surgimento de diferentes aplicações específicas [Vermesan and Friess 2014].

O desenvolvimento de ambientes para IoT é um passo necessário à evolução de tecnologias como Redes Inteligentes de Energia (*Smart Grids*), Sistemas de Transporte Inteligentes (*Intelligent Transportation*) e Cidades Inteligentes (*Smart Cities*). Juntas com a Internet das Coisas, essas soluções fazem parte de uma classe mais genérica de sistemas chamada de Sistemas Ciber-Físicos (*Cyber-Physical Systems*, CPS) [Stojmenovic 2014b].

Esses sistemas são caracterizadas pela colaboração entre elementos computacionais com o intuito de controlar entidades físicas. Em particular, IoT emprega serviços de comunicação convencionais para interligar objetos físicos identificados por endereços baseados na Internet, mas a interconexão de dispositivos através de redes de computadores não é obrigatório em CPS. Além dos objetos físicos, uma arquitetura IoT deve compreen-

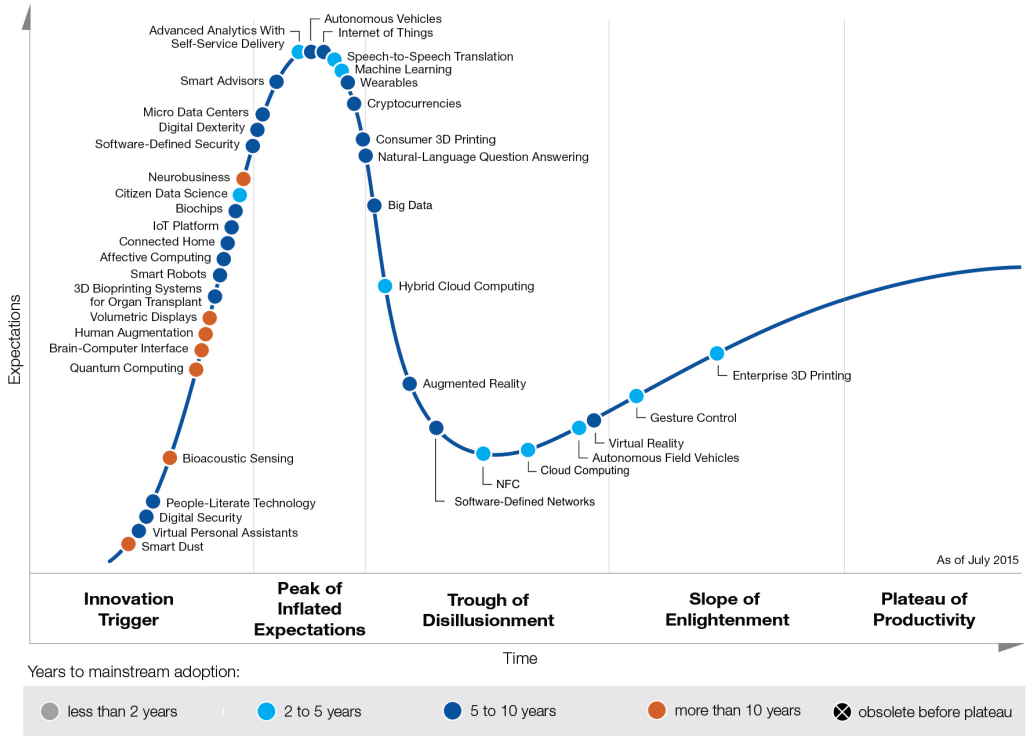


Figura 6.1. O *Hype Cycle* do Gartner divulgado em Julho de 2015 sobre tecnologias emergentes. Fonte: [Gartner Inc. 2015]

der também outros componentes para permitir uma computação ubíqua integrada.

Em [Gubbi et al. 2013] é apresentada uma taxonomia dos elementos necessários à implementação da Internet das Coisas. Os componentes de hardware considerados atômicos como *Radio-Frequency IDentification* (RFID), *Near Field Communication* (NFC) e *Wireless Sensor and Actuator Networks* (WSAN) são responsáveis pela identificação e conexão entre o mundo de objetos reais e sua representação digital. Uma ou mais camadas de software especial ou *middleware* oferecem uma abstração de funcionalidades para lidar com a heterogeneidade e as capacidades limitadas desses elementos, além de prover o gerenciamento e a composição de serviços para uma ampla variedade de aplicações. A Figura 6.2 mostra os componentes IoT sobre uma Arquitetura Baseada em Serviços (*Service Oriented Architecture*, SOA) organizada em diferentes camadas.

Como identificado em [Atzori et al. 2010], o paradigma da Internet das Coisas pode ser concebido sobre três diferentes visões: orientado a "coisas"(sensores), orientada à Internet (*middleware*) ou orientada à semântica (conhecimento). Embora essa delimitação seja necessária pela natureza interdisciplinar do assunto, as soluções em IoT são efetivas apenas no domínio de interseção desses três pontos de vista. Em [Gubbi et al. 2013] é apresentada uma definição unificadora baseada na interconexão de dispositivos embarcados com sensores e atuadores sobre um quadro operacional comum para permitir o desenvolvimento de aplicações inovadoras. Essa abordagem facilita o compartilhamento de informações entre plataformas através de uma estrutura integrada em Nuvem.

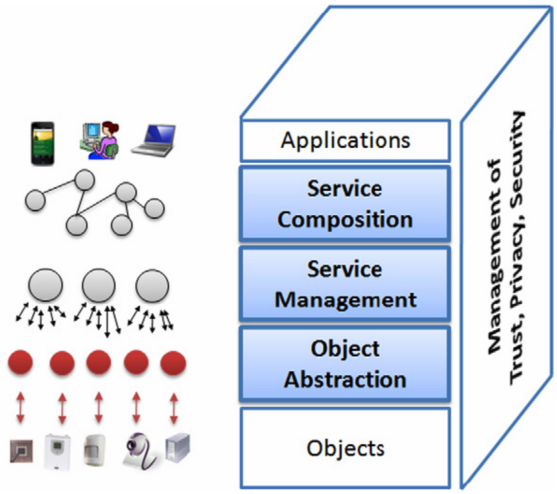


Figura 6.2. Visão geral da arquitetura IoT. Fonte: [Atzori et al. 2010]

6.2.2. Computação em Nuvem

O modelo em Nuvem teve origem na adoção de aplicações que fazem uso de enormes centros de dados com capacidade de suportar serviços Web em larga escala na Internet. Mesmo que a idéia original não seja nova, sua adoção possibilitou o uso eficiente e otimizado dos recursos de hardware e software em ambientes de Tecnologia da Informação (TI). Avanços posteriores em gerenciamento automatizado, técnicas de balanceamento de carga e virtualização permitiram a alocação dinâmica de recursos e o provisionamento elástico de sua infraestrutura aos clientes.

A arquitetura de Nuvem pode ser dividida em quatro camadas [Zhang et al. 2010]: centro de dados (hardware), infraestrutura, plataforma e aplicação. Cada uma delas pode ser vista como um serviço para a camada de cima e como um cliente para a camada inferior. Este arquitetura alcançou popularidade pela oferta da infra-estrutura da Nuvem em três principais modelos de serviço [Zhang et al. 2010]: Software como Serviço (*Software as a Service*, SaaS), Plataforma como Serviço (*Platform as a Service*, PaaS), Infraestrutura como Serviço (*Infrastructure as a Service*, IaaS).

Essa diversidade de clientes e serviços levou a diferentes modelos de desenvolvimento como descritos em [Zhang et al. 2010]: Nuvem Privada (*Private Cloud*), Nuvem Comunitária (*Community Cloud*), Nuvem Pública (*Public Cloud*) e Nuvem Híbrida (*Hybrid Cloud*). Cada tipo de nuvem tem suas próprias vantagens e desvantagens, onde a opção por um modelo depende do cenário específico considerando os diferentes aspectos do negócio, os requisitos dos usuários e os ambientes de nuvem envolvidos.

Seu nível atual de amadurecimento em relação à outras tecnologias pode ser vista através da popularidade em buscadores na Internet. A figura 6.3 mostra a popularidade dos termos *Cloud Computing*, *Big Data* e *Internet of Things* em consultas ao Google nos últimos dez anos [Google Trends 2016]. Os gráficos de popularidade podem ser comparados com o *Hype Cycle* da figura 6.1. Por exemplo, a curva *Cloud Computing* na

figura 6.3 apresenta um contorno similar com a fase de desilusão indicada no *Hype Cycle*, acompanhada por *Big Data* e a Internet das Coisas em seu ápice de expectativas.

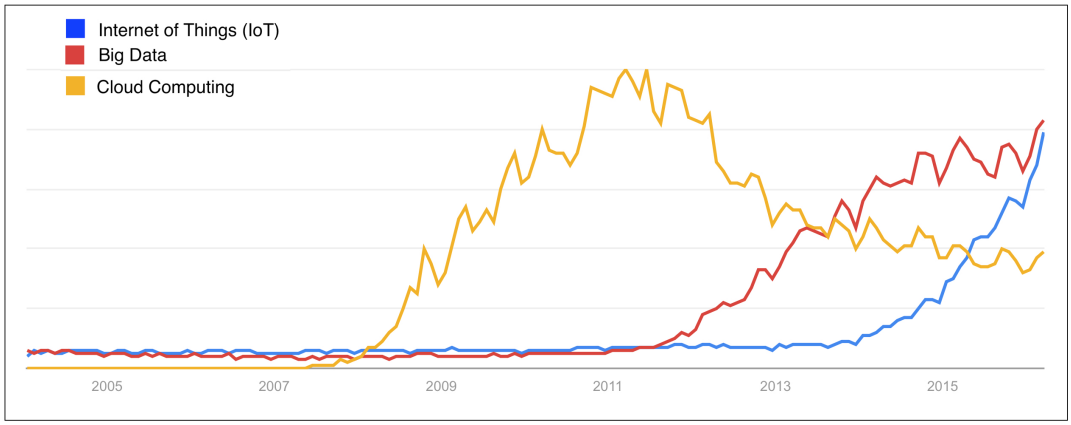


Figura 6.3. Pesquisa do Google sobre a popularidade dos termos *Cloud Computing*, *Big Data* e *Internet of Things*. Fonte: [Google Trends 2016]

Em sua fase de evolução atual, diferentes exemplos dessa tecnologia já se consolidaram através da Internet, onde novos serviços híbridos começam a ser oferecidos por provedores públicos e privados. Embora as empresas mais conservadoras continuem cautelosas quanto à segurança de suas informações, elas migram seus recursos menos críticos para Nuvem buscando usufruir das vantagens técnicas e econômicas desde modelo [Zhang et al. 2015]. Entretanto, a fase de plenitude produtiva desta tecnologia ainda não foi alcançada pela falta de um entendimento completo de suas potencialidades.

Neste sentido, sua agregação com outras tecnologias permitirá que a arquitetura em Nuvem se torne cada vez mais compreendida, difundida e adotada como um importante componente da Internet do Futuro. Em particular, um novo paradigma integrando a Computação em Nuvem com a Internet das Coisas é desafiador e favorável a um grande número de cenários de aplicação, como descrito nos próximos tópicos deste capítulo.

6.2.3. Integração entre Computação em Nuvem e Internet das Coisas

A Computação em Nuvem e a Internet das Coisas são tecnologias que evoluiram de forma independente, baseadas em cenários de aplicação específicos. Embora esses dois paradigmas apresentem definições distintas, pesquisas atuais propõem sua integração com base em aspectos complementares resumidos na Tabela 6.1. Essas soluções são descritas como Nuvem das Coisas (*Cloud of Things*) em [Aazam et al. 2014] e Nuvem IoT (*Cloud IoT*) em [Botta et al. 2016]. Em [Botta et al. 2016], os fatores que motivam essa integração são classificados em três categorias: comunicação, armazenamento e processamento.

- *Motivadores de comunicação*: estão relacionados com o compartilhamento de dados e aplicações através de uma infraestrutura que facilite a conexão entre os elementos e o gerenciamento dos objetos. Neste sentido, as soluções proprietárias em Nuvem oferecem uma forma eficaz para conectar, controlar e gerenciar dispositivos remotos sem restrições de tempo ou localidade através de redes de alta velocidade e aplicativos embutidos em interfaces customizadas na Internet [Rao et al. 2012].

Tabela 6.1. Comparação de aspectos complementares da Computação em Nuvem e IoT.

Característica / Paradigma	IoT	Nuvem
Modelo de computação	distribuído ou pervasivo	centralizado
Disponibilidade de acesso	local ou limitado	global ou ubíquo
Natureza dos componentes	objetos físicos	recursos virtuais
Capacidade de processamento	limitada	virtualmente ilimitada
Capacidade de armazenamento	limitada ou nenhuma	virtualmente ilimitada
Função da Internet	ponto de convergência	meio de prover serviços
Análise de dados	análise em tempo real	análise de *Big Data*

- *Motivadores de armazenamento*: buscam compensar as restrições de espaço para arquivos de dados nos dispositivos IoT através de um serviço de memória não volátil em larga escala e sob demanda, baseada na virtualização ilimitada de recursos [Rao et al. 2012]. A Internet das Coisas envolve muitas fontes ou objetos que geram um fluxo de dados com características de volume (quantidade), variedade (tipos de dados) e velocidade (frequência) típicas de *Big Data* [Zikopoulos et al. 2011]. Uma vez depositados na Nuvem, os dados podem ser protegidos do acesso indevido por níveis configuráveis de segurança [Dash et al. 2010] e podem ser manipulados de maneira uniforme [Zaslavsky et al. 2013] ou compartilhados usando uma API (*Application Programming Interface*) bem definida [Fox et al. 2012].

- *Motivadores de processamento*: dizem respeito às limitações computacionais dos dispositivos IoT na execução local de algoritmos complexos considerando suas restrições de energia [Yao et al. 2013]. Em ambientes IoT, os dados coletados são usualmente transmitidos para outros elementos mais poderosos, onde sua agregação e processamento é factível. Transferir essa responsabilidade para uma plataforma de Nuvem permite economizar a energia dos dispositivos IoT acessando serviços externos sob demanda. Além disso, seu processamento virtualmente ilimitado possibilita a realização de análise de dados [Dash et al. 2010] [Rao et al. 2012] e o controle de eventos complexos [Rao et al. 2012].

Outros motivadores em [Botta et al. 2016] são apontados como transversais e têm implicações em todas as categorias. Por exemplo, o modelo em Nuvem vem abordando questões que também precisam ser tratadas em IoT. As soluções existentes para problemas de heterogeneidade, escalabilidade, interoperabilidade, flexibilidade, confiabilidade, eficiência, disponibilidade e segurança são motivadores transversais à integração desses dois paradigmas. Além disso, agregando o fluxo de dados dos objetos em uma infraestrutura unificada torna possível uma rápida configuração e integração de novos elementos. Essa fator incentiva e facilita a implantação dos serviços, proporcionando meios para aumentar receitas e reduzir os riscos envolvidos [Zaslavsky et al. 2013].

Entretanto, a Computação em Nuvem também pode se beneficiar da Internet das Coisas para ampliar seus limites, fornecendo suporte a grande número de situações do mundo real. A arquitetura em Nuvem IoT deve fornecer uma camada intermediária para abstrair a complexidade e prover as funcionalidades necessárias entre os objetos e as aplicações. Uma arquitetura típica é descrita em [Taivalsaari and Mikkonen 2015] onde seus elementos de mais alto nível estão representados na Figura 6.4.

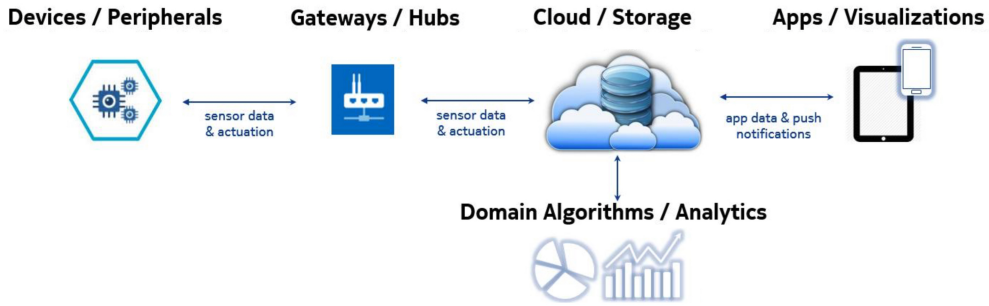


Figura 6.4. Arquitetura Comum em Nuvem IoT. Fonte: [Taivalsaari and Mikkonen 2015]

Nesta arquitetura comum, os objetos ou dispositivos embarcados com sensores e ou atuadores interagem para ativar ações ou submeter dados em resposta aos pedidos de outros componentes de capacidade superior em uma rede de sensores. Os *gateways* ou *hubs* são dispositivos de acionamento, coleta e transferência de dados a partir dos dispositivos periféricos com sensores e atuadores para Nuvem. Usando diferentes protocolos de comunicação (com ou sem fio) permitem uma conexão segura entre diferentes redes físicas e podem apoiar tarefas como armazenamento temporário, *cache* de dados, pré-processamentos, descoberta de serviços, geo-localização e cobrança. Em algumas soluções, os dispositivos embarcados são capazes de receber comandos de atuação ou enviar dados de sensoriamento diretamente para Nuvem sem a necessidade de *gateways* dedicados, ou agir como *gateways* intermediários para outros dispositivos.

Como discutido anteriormente nos motivadores de integração, a Nuvem possui um papel central nesta arquitetura. Uma de suas funções envolve a manutenção do registro com informações ou metadados sobre os objetos gerenciados no sistema. No caso mais simples, o registro pode ser um arquivo em um banco de dados contendo IDs (identificadores) ou URL (*Uniform Resource Locator*) dos dispositivos. Porém, normalmente é um componente abstrato que fornece uma API de programação independentemente da solução de armazenamento subjacente. Assim, os desenvolvedores podem obter e apresentar as informações em diferentes interfaces (móvel, web, etc.) para os usuários.

Outra função importante é a aquisição e armazenamento de dados, onde as soluções podem envolver sistemas escaláveis com replicação massiva e tolerância a falhas. Devido ao crescente aumento no número de dispositivos associados a IoT, as tecnologias em Nuvem podem ser empregadas no armazenamento do enorme volume de dados gerados por esses objetos. Requisitos de latência ou atraso para acesso aos dados históricos ou informações deles extraídas podem variar dependendo da aplicação.

As funções de análise e visualização de informações buscam examinar, relacionar e transformar os dados adquiridos através dos sensores a fim de descobrir e apresentar informações úteis os usuários ou ao próprio sistema. Devido ao volume, variedade e velocidade de informações geradas pelos dispositivos, diferentes algoritmos de aprendizado de máquina e mineração de dados como redes neurais, algoritmos genéticos ou árvores de decisão podem ser empregados na análise dos dados gerados por esses objetos [Bonomi et al. 2012]. A integração das tecnologias de Nuvem com a análise de *Big Data* vem per-

mitindo um ciclo constante de inovação favorável ao desenvolvimento das plataformas voltadas à IoT [Mineraud et al. 2015].

De forma similar à análise de dados, funções que permitam a programação dos objetos bem como a configuração de ações baseadas em informações históricas ou instantâneas sobre dados de sensores devem ser definidas de forma ampla e segura nesta arquitetura. A Nuvem é um ambiente atrativo para o desenvolvimento de aplicações devido aos seus modelos de programação em alto nível que facilitam o desenvolvimento de serviços em larga escala. A oferta de interfaces integradas e confiáveis para programação remota dos elementos atuadores é fundamental para assegurar uma visão ubíqua em IoT uma vez que sua execução envolve diferentes dispositivos móveis e sensores.

Na medida que novos tipos de informações, recursos, objetos e pessoas são conectadas nesta arquitetura, os usuários espalhados por todo o mundo devem fazer parte rapidamente da Internet de Coisas. A adoção de um modelo em Nuvem para IoT permite a criação de novos paradigmas de serviço inteligentes com base na capacidades de seus componentes para lidar com uma série de cenários futuros. Descritos genéricamente em [Duan et al. 2015] como XaaS (*Everything as a Service*), essas propostas incorporam diferentes aspectos de "coisas"como serviço ou *Things as a Service* (TaaS) [Christophe et al. 2011] [Mitton et al. 2012] [Distefano et al. 2012]. Alguns exemplos e suas referências como identificado em [Botta et al. 2016] estão resumidos na Tabela 6.2 .

Esta seção abordou as principais motivações que impulsionam a integração do modelo em Nuvem com a Internet das Coisas. Porém, restrições inerentes a sua arquitetura dificultam cenários de aplicação específicos envolvendo um número crescente de objetos em ambientes densamente distribuídos. Para isso, descrevemos na próxima seção como algumas propostas recentes buscam estender os recursos computacionais disponíveis na Nuvem para as extremidades da rede visando apoio às aplicações em IoT. Exemplos de soluções e plataformas serão descritas a seguir na seção 6.3.

6.2.4. Tecnologias Emergentes de Nuvem IoT

Apesar das potenciais vantagens de modelo baseado em Nuvem para Internet das coisas, alguns cenários não são favoráveis a sua aplicação. Os sistemas de Computação em Nuvem são altamente centralizados, onde maioria da computação ocorre em poucos e grandes centros de dados espalhados pelo mundo. Embora esta abordagem ofereça benefícios, ela tem um custo elevado em termos de comunicação e consumo de energia.

Em ambientes IoT onde os dispositivos estão geograficamente próximos uns dos outros, seria ineficiente transmitir todos os dados de sensoriamento para núcleos de processamento distantes e esperar por comandos a partir de um centro remoto para dispositivos atuadores individuais. Nesse cenário, eventuais sobrecargas e atrasos muito peculiares na Internet tornariam as soluções IoT com requisitos de baixa latência impraticáveis. Por exemplo, aplicações para Internet Tátil (*Tactile Internet*) [Simsek et al. 2016] requerem conexões ultra-confiáveis e com latência de 100 ms, 10 ms e 1 ms para realizar respectivamente experiências auditivas, visuais e manuais remotas em tempo real.

As soluções que requerem uma latência muito baixa a nível de aplicação são desafiadoras na Internet. Uma partícula viajando a velocidade da luz (186,3 milhas por

Tabela 6.2. Novos paradigmas de serviço em Nuvem IoT.

XaaS (por extenso)	Referências	Descrição
TaaS (Things as a Service)	[Distefano et al. 2012] [Mitton et al. 2012] [Christophe et al. 2011]	abstração de recursos heterogêneos com semântica similar a dos objetos
SaaS (Sensing as a Service)	[Zaslavsky et al. 2013] [Rao et al. 2012] [Dash et al. 2010]	acesso ubíquo a dados de sensores
SAaaS (Sensing and Actuation as a Service)	[Rao et al. 2012]	implementação da lógica de controle e automação
SEaaS (Sensing Event as a Service)	[Rao et al. 2012] [Dash et al. 2010]	envio de mensagens desencadeadas por eventos em sensores
SenaaS (Sensor as a Service)	[Zaslavsky et al. 2013]	gerenciamento ubíquo de sensores remotos
DBaaS (Database as a Service)	[Zaslavsky et al. 2013]	gerenciamento ubíquo de banco de dados
DaaS (Data as a Service)	[Zaslavsky et al. 2013]	acesso ubíquo a qualquer dado
EaaS (Ethernet as a Service)	[Zaslavsky et al. 2013]	conectividade em nível de camada 2 para dispositivos remotos
IPMaaS (Identity and Policy Management as a Service)	[Zaslavsky et al. 2013]	gerenciamento de políticas de acesso e identidade

milésimo de segundo) levaria aproximadamente 22,5 ms para viajar ida e volta a menor distância de costa a costa (2092 milhas) dos Estados Unidos. Mesmo sem considerar atrasos de *buffering*, filas, ou qualquer tipo de processamento na Nuvem, esse tempo de latência inviabilizaria diferentes tipos de aplicações de tempo real.

Outra questão é o aumento do número dos objetos, que deve gerar uma enorme quantidade de dados produzidos nas extremidades da Internet. Em 2014, o volume de dados gerados por dispositivos IoT ultrapassou 200 exabytes, sendo estimado um total anual de 1,6 zettabytes em 2020 [ABI Research 2015]. Atualmente, apenas uma pequena percentagem dos dados produzidos por esses elementos são enviados e ou armazenados em Nuvem. A maioria é processado ou armazenado localmente e não estão acessíveis para aplicação de técnicas como análise de dados ou *Big Data* [ABI Research 2015].

Agravando este cenário, o custo médio de largura de banda vem diminuindo a uma taxa inferior quando comparado ao custo médio de processamento ou de armazenamento [Nielsen Norman Group 2014]. O aumento do poder computacional torna os dispositivos miniaturizados cada vez mais acessíveis e produtivos enquanto a capacidade de enviar dados a partir da borda da Internet para a Nuvem enfrenta atrasos causados pelo crescimento impulsivo da concorrência aos recursos de comunicação disponíveis.

Essa tendência para concentração dos dados nas bordas da rede é uma realidade contemporânea e estimulada pelo atual crescimento da Internet das Coisas. Diferentes técnicas que minimizem a quantidade de dados enviados para Nuvem através do processamento local em elementos periféricos serão fundamentais na redução dos custos envolvidos e no tempo de resposta das aplicações. Uma vez que nem todos os dados estarão localizados no núcleo da arquitetura de Nuvem IoT, os aplicativos também serão atraídos para borda da rede pelo efeito crescente da gravidade de dados [Fritsch and Walker 2014].

A execução de aplicações nas extremidades da rede passa a ser um interessante estímulo para ampliar as fronteiras do modelo de Nuvem IoT visando aproveitar o poder de processamento, armazenamento e comunicação atual dos diferentes tipos de dispositivos móveis inteligentes que são conduzidos por pessoas em todos os lugares do mundo. Por exemplo, realidade virtual, sensoriamento e navegação são exemplos de aplicações sofisticadas que podem se beneficiar de recursos locais oferecidos aos usuários.

Como as aplicações dos usuários são executadas em redes sem fio ou redes de rádio celular, o conceito de borda de rede envolve os provedores de acesso à Internet (*Internet Service Provider*, ISP). Por exemplo, o código de uma aplicação de realidade aumentada poderia ser dividida em três partes: o aplicativo cliente executado no celular do usuário, os serviços remotos executados a partir do servidor em Nuvem e os serviços locais executados junto (ou em nível acima) da estação base de acesso do usuário.

Quando a execução do código da parte servidor está localizado na borda da rede, a aplicação pode fazer uso de informações em tempo real para tornar mais inteligente a entrega do serviço ao usuário. Por exemplo, informações de congestionamento e largura de banda de acesso concedida através de ISPs podem ser usadas por aplicativos que utilizam a plataforma *Netflix Open Connect* [Netflix 2016] para definir a qualidade do fluxo de vídeo suportado ou seu redirecionamento para o servidor de borda mais próximo ao cliente, como parte de uma solução em Nuvem para entrega de conteúdo sob demanda.

Entretanto, uma infraestrutura em Nuvem plenamente extensível para borda da rede depende da cooperação entre diferentes atores como prestadoras de serviços em Nuvens púbicas e privadas, desenvolvedores em Internet das Coisas, provedores de acesso à Internet e diferentes organizações de padronização. Atualmente, a indústria de telecomunicações está passando por uma grande revolução técnica caracterizada pela presença de tecnologias facilitadoras exemplificadas nas propostas abaixo:

- *Redes programáveis* - Em [Manzalini and Crespi 2016] é apresentada uma plataforma de telecomunicações que emprega tecnologias de Virtualização das Funções da Rede (*Network Functions Virtualization*, NFV) e Redes Definidas por Software (*Software Defined Network*, SDN) de modo a facilitar a implementação de serviços em Nuvem através da borda da rede. O paradigma SDN separa os planos de controle (inteligência) e comunicação de dados (encaminhamento) em funções distintas, onde os nós da rede seguem o caminho de comunicação obtido pela consulta a um servidor centralizado. Porém, em ambientes com diferentes domínios, esse serviço de controle centralizado pode precisar de uma implementação distribuída.

- *Fatiamento (slicing) de rede* - Em [Shimojo et al. 2015], é proposta uma arquitetura em redes móveis para o fornecimento e operação de serviços heterogêneos baseado no fatiamento virtual da infraestrutura de telecomunicações física disponível (canais, redes, hardware, etc.) de forma customizada e configurada para fins específicos ou arrendatários, e atendendo diferentes requisitos dos clientes.

- *Interface de rádio 5G* - Em [Simsek et al. 2016] esta tecnologia é descrita como um padrão ainda em desenvolvimento que será supostamente flexível para possibilitar uma variedade de novos cenários de uso em redes móveis, incluindo suporte a uma alta largura de banda e baixa latência.

Essas tecnologias podem oferecer suporte as soluções em Nuvem IoT a partir da perspectiva de redes de comunicação, mas sozinhas não são suficientes para abordar todos os problemas relacionados à extensão dessa arquitetura para borda da rede. Outros paradigmas emergentes da integração do modelo em Nuvem com a Internet das Coisas empregam terminologias distintas para representar conceitos semelhantes, onde recursos de computação são localizados próximos dos usuários visando superar limitações como mobilidade, localidade e baixa latência. A popularidade dos termos mais empregados são comparadas na Figura 6.5 e seus significados são descritos logo a seguir.

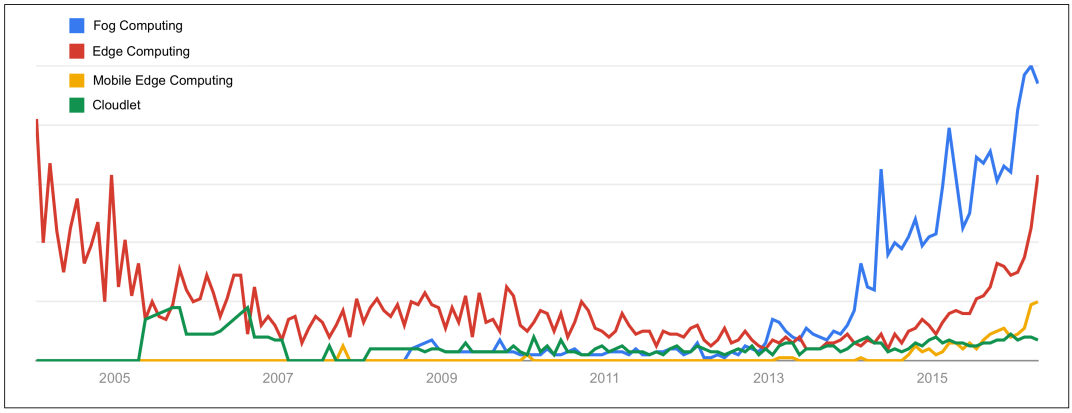


Figura 6.5. Paradigmas emergentes do modelo Nuvem IoT. Fonte: [Google Trends 2016]

- Computação nas Bordas (*Edge Computing*) - no gráfico da Figura 6.5 é o termo mais usado e genérico para designar tecnologias em borda de rede. Entretanto, é usado em propostas que nem sempre envolvem conceitos da Computação em Nuvem ou Internet das Coisas como autenticação de consultas em banco de dados distribuídos e replicação de *caches*. O termo *Edge Cloud Computing*, embora pouco usado, é mais adequado por ressaltar o relacionamento com as tecnologias de Nuvem.

- Mini-nuvens (*Cloudlets*) - pesquisadores da *Carnegie Mellon University* (CMU) [Elijah 2016] apresentaram o conceito em [Satyanarayanan et al. 2009] como uma camada intermediária entre dispositivos móveis e o centro de computação em Nuvem. Este trabalho pioneiro disponibilizou sistemas de código aberto em [Github 2016]. Enquanto [Gao et al. 2015] e [Satyanarayanan et al. 2015] discutem sua utilidade, [Ha and Satyanarayanan 2015] apresenta uma visão técnica detalhada. Como o termo é usado em outras áreas, a curva no gráfico na Figura 6.5 pode mostrar uma popularidade maior do que apenas no domínio de tecnologias de Nuvem.

- Computação Móvel em Nuvem (*Mobile Cloud Computing*, MCC) - é definida em [Dinh et al. 2013] como uma infra-estrutura onde o processamento e o armazenamento de dados são movidos dos dispositivos móveis para as plataformas em Nuvem. Esses dispositivos móveis não precisam possuir uma configuração avançada de hardware visto que as aplicações são executadas remotamente na Nuvem e acessadas através de conexões sem fio. A interação dos usuários com o sistema acontece por meio de interfaces leves ou navegadores *web* instalados nos clientes.

- Computação Móvel nas Bordas (*Mobile Edge Computing*, MEC) - definida pelo *European Telecommunications Standards Institute* (ETSI) através de uma especificação para indústria em [ETSI ISG MEC 2015]. Envolve o desenvolvimento de uma arquitetura e uma série de APIs padronizadas para o suporte da computação em Nuvem através das estações base nas redes móveis de rádio.

- Micro Centro de Dados (*Micro Datacenters*, MDC) - a Microsoft anunciou em [NetworkWorld 2015] seu emprego como uma extensão aos centros de dados Microsoft Azure para tratar dos custos envolvidos com as tecnologia de Nuvem [Greenberg et al. 2008], optimizar o desempenho dos pequenos dispositivos e melhorar a performance das aplicações dirigidas a Internet das Coisas.

- Computação em Névoa (*Fog Computing*) - é considerada como uma extensão não-trivial da computação em Nuvem [Bonomi et al. 2012]. Em [Vaquero and Rodero-Merino 2014] é oferecida uma definição abrangente, com ênfase em algumas propriedades importantes como a predominância de acesso sem fio, heterogeneidade, distribuição geográfica, ambiente de execução, etc.

Entretanto, não existe uma nítida distinção entre essas propostas que parecem convergir sobre os seguintes pontos identificados em [Klas 2016]:

- *Mini-centros de processamento de dados* - introduzem uma versão em miniatura de um ambiente de Computação em Nuvem com suporte a virtualização de recursos que podem ser mantidos pelas companhias de telecomunicações e outras empresas consorciadas. Dependendo da proposta acima, esses elementos podem ser referenciados como *micro datacenters*, *cloudlets*, *fog nodes* ou servidores MEC.

- *Distribuição de recursos em larga escala* - promovem a distribuição de pequenas Nuvens em centenas ou milhares de pontos estratégicos de um estado ou país para fornecer recursos computacionais aos clientes. Sua localização dependerá de fatores como economia de energia ou requisitos de latência das aplicações, distribuídas em uma arquitetura com no mínimo três camadas como mostrado na Figura 6.6. Pontos prováveis de instalação são nas estações base, em comutadores de agregação ou nas centrais de suporte de rede das empresas de telecomunicações.

- *Infra-estrutura e plataforma como Serviço*: oferecem uma infra-estrutura integrada em pequenas Nuvens seguindo os modelos IaaS e PaaS descritos na seção 6.2.2. Nesta plataforma, aplicações podem ser implementadas de maneira bastante ágil usando ambientes de Máquina Virtual (*Virtual Machine*, VM). O modelo de plataforma como serviço favorece a migração de VMs em tempo de execução e sob demanda entre as pequenas Nuvens para tratar questões de mobilidade dos clientes.

- *Serviços Especiais* - oferecem serviços exclusivos em algumas pequenas Nuvens usando informações apenas presentes na rede local ou em suas proximidades. Por exemplo, uma pequena Nuvem pode hospedar um serviço que fornece acesso as estimativas de tráfego na vizinhança. Essas informações podem ser utilizadas pela aplicação em tempo real para ajustar as rotas iniciais transmitidas aos usuários.

Dentre os termos analisados na Figura 6.5 é possível notar que a Computação em Névoa vem crescendo constantemente desde a sua adoção no final de 2012 pela Cisco [Bonomi et al. 2012]. Os tópicos a seguir abordam esse paradigma, que desponta como uma solução integrada para estender os recursos da Computação em Nuvem em direção a borda da rede buscando cumprir requisitos não atendidos por um modelo clássico centralizado.

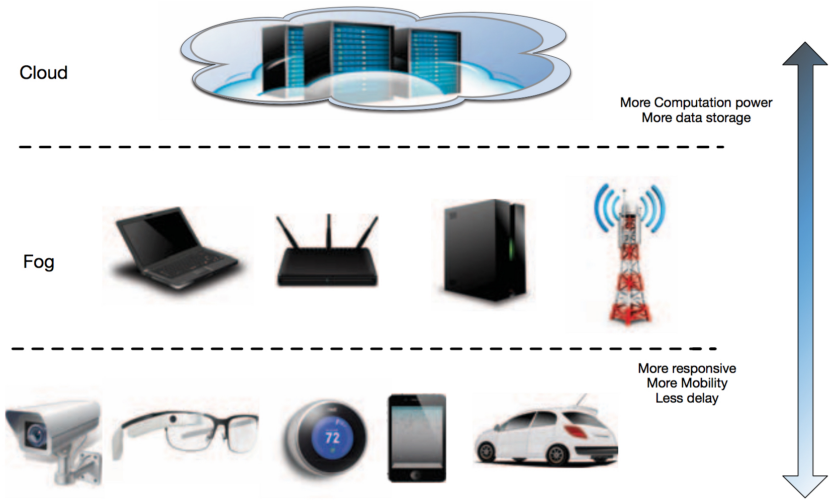


Figura 6.6. Arquitetura em três camadas: dispositivos inteligentes e sensores, Névoa/*cloudlets*/MEC e a Computação em Nuvem. Fonte: [Yi et al. 2015c]

6.2.5. Caracterização da Computação em Névoa

A computação em Névoa é definida em [Bonomi et al. 2012] como uma plataforma altamente virtualizada que fornece serviços de computação, armazenamento e comunicação entre os dispositivos finais e os centros de dados tradicionais de computação em Nuvem, localizada normalmente, mas não exclusivamente, na borda da rede. Essa definição implica em uma série características que tornam a Névoa uma extensão não-trivial da Nuvem, como descritas em [Bonomi et al. 2012] a seguir.

- *Reconhecimento de localidade e baixa latência* - a origem da computação em névoa pode ser atribuída às propostas anteriores envolvendo pontos de acesso com suporte a serviços sofisticados na borda da rede, incluindo aplicações com requisitos de baixa latência como jogos, *streaming* de vídeo e realidade aumentada.

- *Distribuição geográfica* - Em nítido contraste com a computação mais centralizada da Nuvem, os serviços e aplicações orientadas para Névoa demandam implantações amplamente distribuídas. Por exemplo, a Névoa vai desempenhar um papel ativo no fornecimento de *streaming* de alta qualidade para veículos em movimento, através de servidores proxies e pontos de acesso posicionados ao longo das estradas.

- *Suporte a redes de sensores em larga escala* - o monitoramento do ambiente através das redes de sensores, o controle de tráfego entre veículos conectados e as redes de

energia inteligentes (*Smart Grids*) em [Satyanarayanan et al. 2015] são exemplos de sistemas inerentemente distribuídos que exigem o controle e a oferta dos recursos de processamento, armazenamento e comunicação em uma escala Máquina-a-Máquina (*Machine-to-Machine*, M2M) [Stojmenovic 2014b].

- *Grande número de nós* - é uma consequência da ampla distribuição geográfica como evidenciada pelas tecnologias de redes de sensores em geral, e pela aplicação *Smart Grid* em particular. Os serviços são executados pelos nós presentes na névoa como parte de uma aplicação em nuvem distribuída. Quando possível, esse objetivo não é simples de alcançar em arquiteturas de hiperescala como a Internet.

- *Suporte a computação móvel* - em muitas aplicações de névoa é essencial uma comunicação direta com os dispositivos associados para permitir o suporte às diferentes técnicas que podem ser empregadas na computação móvel, como por exemplo o protocolo LISP 1, que desassocia a identidade utilizada por um cliente da sua identidade local, e exige um sistema de diretório distribuído.

- *Interações em tempo real* - As principais aplicações em névoa envolvem interações em tempo real, em vez do processamento em lote. Como a Névoa está localizada longe dos principais centros de processamento em Nuvem, é necessário a disponibilização e o gerenciamento dos recursos locais necessários para auxiliar o cumprimento dos requisitos de tempo de execução das aplicações.

- *Predominância do acesso sem fio* - Para os dispositivos IoT algum tipo de protocolo de comunicação sem fio como RFID, Bluetooth, ZigBee, Wi-Fi ou LTE é a única forma possível de conexão em rede. Os nós presentes na Névoa devem oferecer serviços especiais que só podem ser exigidos no contexto da Internet das coisas.

- *Heterogeneidade* - as Nuvens são geralmente ambientes fechados, que utilizam componentes de hardware de um mesmo fornecedor ou ambientes e linguagens de programação proprietárias. Pela sua abrangência e escopo, os dispositivos na Névoa podem ser oferecidos por diversos fabricantes, empregar diferentes ambientes e envolver variados desenvolvedores, linguagens e protocolos.

- *Interoperabilidade e federação* - o suporte contínuo e integrado de certos serviços requer a cooperação de diferentes provedores, onde *streaming* é um bom exemplo. Assim, os componentes de névoa devem ser capazes de interagir de forma orquestrada e os serviços devem ser federados através de diferentes domínios.

- *Análise de dados em tempo real* - Interagindo com a Nuvem e perto das fontes de dados, a Névoa esta bem localizada para desempenhar um papel significativo na ingestão e processamento de dados em *Big Data* com restrições de tempo real.

A Computação em Nevoa envolve a execução de aplicações sobre elementos distrubuídos nas camadas entre os dipositivos sensores e a Nuvem. Elementos como *gateways* inteligentes, roteadores e dispositivos de névoa dedicados podem oferecer recursos de processamento e armazenamento para permitir a extensão dos serviços de Nuvem

até a borda da rede. A Figura 6.7 mostra uma arquitetura de referência para a Computação em Névoa apresentada em [Dastjerdi et al. 2016]. Na parte inferior desta arquitetura encontram-se os objetos inteligentes, redes de sensores e atuadores, bem como dispositivos de borda e os *gateways*. Esta camada inclui aplicativos que podem ser instaladas nos dispositivos finais para ampliar sua funcionalidade. Os dispositivos nesta camada podem usar a camada seguinte para a comunicação com outros dispositivos ou com a Nuvem.

A camada de Rede deve favorecer também a conexão de outros elementos, não necessariamente sensores ou atuadores, conectados através de tecnologias de rede com ou sem fio. Além disso, esta camada pode prover acesso a recursos de rede virtualizados como instâncias de névoa através de elementos inteligentes, capazes de processar e armazenar temporariamente dados coletados pelos *gateways* sobre dispositivos IoT da camada inferior. Estes dispositivos de névoa também são responsáveis por filtrar e enviar informações para a Nuvem em uma base periódica de tempo.

Acima da camada de Rede são executados os serviços que oferecem suporte ao processamento de tarefas voltadas a Internet das Coisas para aplicações que precisam do auxílio de recursos virtualmente ilimitados disponíveis na Nuvem. No topo da camada de Nuvem reside o software de gerenciamento de recursos que coordena de forma global a infraestrutura e oferece qualidade de serviço para as aplicações em Névoa. Finalmente, na camada superior estão as aplicações que utilizam a infraestrutura de Computação em Névoa para fornecer soluções inovadoras e inteligentes para os usuários finais.

A camada de Gerenciamento de Recursos Definidos por Software implementa diferentes serviços de *middleware* para otimizar o uso dos recursos de Nuvem e Névoa em benefício das aplicações. O objetivo destes serviços é reduzir a carga de utilização da Nuvem ao mesmo tempo que melhora o desempenho das aplicações, transferindo a execução de tarefas para nós da névoa e oferecendo níveis aceitáveis de latência. Isto pode ser alcançado pelo trabalho conjunto de diferentes serviços descritos a seguir:

- *Localização de Fluxo e Tarefas* - mantém informação sobre o estado dos recursos disponíveis na Nuvem, Névoa e na rede a partir de consultas ao serviço de Monitoramento visando identificar os melhores candidatos para receber tarefas e para suportar os fluxos de execução. Este componente se comunica com o serviço de Provisionamento de Recursos para indicar o número atual de fluxos e tarefas alocadas, o que pode desencadear uma nova rodada de alocações quando a demanda pelos recursos for considerada elevada.

- *Base de Dados de Conhecimento* - armazena informações históricas sobre a demanda de aplicações e recursos, podendo ser fornecidas a outros serviços para auxiliar o processo de tomada de decisão.

- *Previsão de Desempenho* - consulta informações na Base de Conhecimento para estimar o desempenho dos recursos disponíveis. Esta informação é usada pelo serviço de Provisionamento de Recursos para decidir a quantidade de recursos que serão disponibilizados em momentos quando existe um grande número de tarefas e fluxos alocados, ou quando o desempenho não é satisfatório.

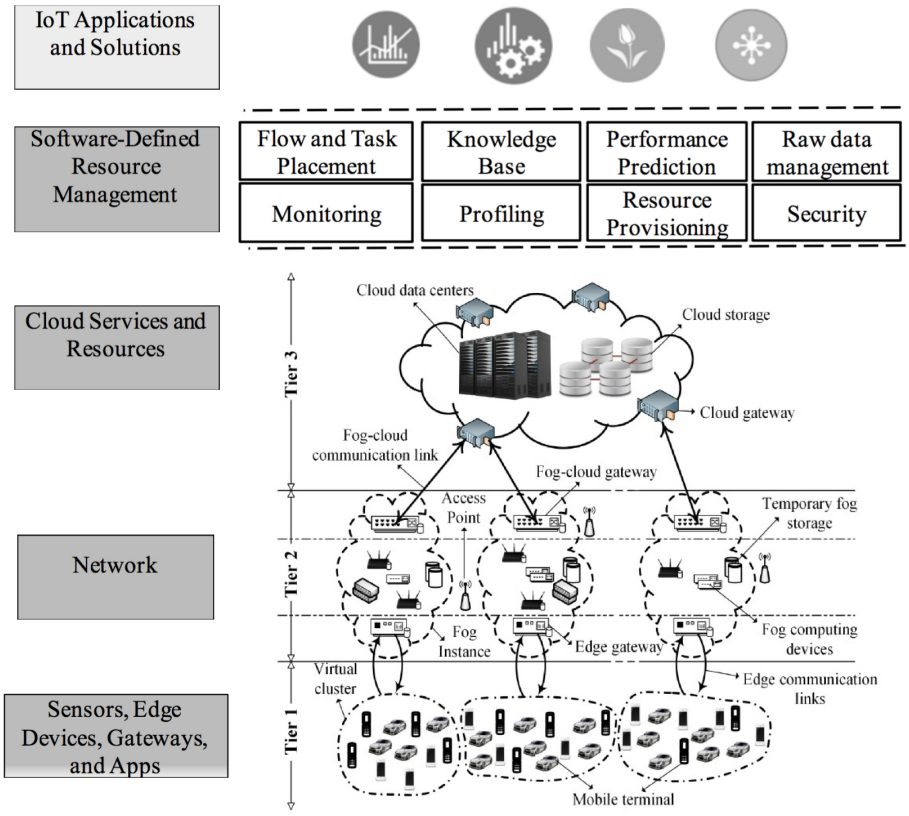


Figura 6.7. Arquitetura de referência para Computação em Névoa. Fonte: [Dastjerdi et al. 2016] e [Sarkar et al. 2015]

- *Gerenciamento de Dados* - oferece visões sobre os dados para outros serviços através do acesso direto às fontes e arquivos históricos. Essas visões podem ser obtidas por meio de consultas simples em SQL ou através de processamento mais complexos envolvendo técnicas de *Big Data* como *MapReduce*. No entanto, o método específico empregado na geração das visões é abstraído para os outros serviços.

- *Monitoramento* - mantém o controle do desempenho e do estado das aplicações, recursos e outros serviços, fornecendo essas informações conforme solicitado.

- *Gerenciador de Perfis* - estabelece perfis de recursos e aplicações baseadas em informações obtidas a partir da Base de Conhecimento e Monitoramento de serviços.

- *Provisionamento de Recursos* - é responsável pela aquisição de recursos na nuvem, névoa e na rede para hospedar as aplicações. Esta alocação é dinâmica, uma vez que o número de aplicações hospedadas e seus requisitos mudam ao longo do tempo. A decisão sobre os recursos é feita baseada nos requisitos de latência, nas credenciais gerenciadas pelo Serviço de Segurança, e nas informações fornecidas por outros serviços como Gerenciador de Perfis, Previsão de Desempenho e Monitoramento. Por exemplo, o componente transfere as tarefas com requisitos de baixa latência

para a borda da rede logo que os recursos apropriados ficam disponíveis.

- *Segurança*: fornece serviços de autenticação, autorização e criptografia para acesso e execução de serviços e aplicações, estendido a todos os elementos da névoa.

É importante destacar que todos os elementos e serviços descritos são apenas de referência. As pilhas de protocolo e aplicações para Névoa podem ser desenvolvidas sem o uso de todos os elementos descritos, ou podem integrar outros componentes e serviços não apresentados na Figura 6.7 como aqueles relacionados na plataforma da Figura 6.11.

6.3. Aplicações e Plataformas de Computação em Névoa

Iniciativas para o emprego de soluções em Névoa devem surgir em diferentes setores de domínio público e privado. Nesta seção, buscaremos oferecer uma visão dessas aplicações em diferentes áreas e examinar suas plataformas sobre um ponto de vista arquitetural, bem como identificar a utilização de técnicas como virtualização, análise de dados, *caching* e segurança em redes na organização e implementação desses sistemas.

6.3.1. Aplicações em Computação em Névoa

As aplicações sobre a Internet das Coisas podem ser tão ou mais variadas quanto os dispositivos nela empregados. Elas possuem em comum a análise de dados em tempo real de "coisas" conectadas, podendo promover ações sobre seus elementos. Cada ação executada pode envolver uma comunicação Máquina-a-Máquina (*Machine-to-Machine*, M2M) [Stojmenovic 2014b] ou uma interação Homem-a-Máquina (*Human Machine Interface*, HMI) [Bonomi et al. 2012]. Exemplos incluem o acender de uma lâmpada, o acionamento de um motor, o fechamento de uma porta, a alteração das configurações de equipamentos, o acionamento dos freios de um carro, o movimento ou foco de uma câmera de vídeo, a abertura de uma válvula em resposta a uma leitura de pressão, etc..

Considerando as características descritas na seção 6.2.5, existe um conjunto de aplicações em Nuvem IoT que podem ser apoiadas pela computação em Névoa. Este tópico será então subdividido em áreas onde o atendimento desses requisitos contribuem para melhorar as propostas e soluções. Serão discutidos principalmente cenários que são fundamentais, motivadores e beneficiários do conceito de Computação em Névoa.

6.3.1.1. Redes de Sensores e Atuadores

Os nós tradicionais em redes de sensores sem fio (*Wireless Sensor Networks*, WSNs), comumente chamados *motes* [Kashi and Sharifi 2013], são projetados para trabalhar com níveis de energia extremamente baixos buscando prolongar a vida da bateria ou até mesmo acumular energia do ambiente quando possível [Rahimi et al. 2003].

A maioria dos WSNs (*Wireless Sensor Nodes*) podem ser distribuídos em amplas áreas geográficas, requerendo baixa largura de banda, baixo consumo de energia, baixo poder de processamento e pouca capacidade de memória. Atuando como fontes de dados unidirecionais para *gateways* estáticos chamados *sinks*, executam tarefas simples de processamento e encaminhamento de dados. O sistema operacional de código

aberto [TinyOS 2016] é um padrão usado nesses tipos de dispositivos, e vem provando sua utilidade em uma variedade de cenários envolvendo coleta dados em diferentes ambientes (temperatura, humidade, quantidade de precipitação, intensidade de luz, etc.).

Devido às limitações dos WSNs, são propostas diferentes configurações para atender aos requisitos das aplicações [Kashi and Sharifi 2013]: múltiplos *sinks*, *sinks* móveis, múltiplos *sinks* móveis e sensores móveis. No entanto, em aplicações que exigem mais do que apenas funções de detecção e rastreamento é necessário o uso de atuadores exercendo ações físicas como abertura, fechamento, movimento, foco, etc. Os atuadores, recebendo comandos dos *sinks*, podem controlar um sistema ou o próprio processo de medição, ampliando o emprego de *Wireless Sensor and Actuator Networks* (WSANs).

Com o uso de atuadores, o fluxo de informação passa a ser bidirecional: dos sensores para o *sink* e do nó controlador para os atuadores. Essas soluções tornam-se sistemas de ciclo fechado em que os potenciais problemas de estabilidade e comportamentos oscilatórios não podem ser ignorados [Bonomi et al. 2012]. A latência e o *jitter* são questões dominantes em sistemas que requerem uma resposta rápida.

As características da Computação em Névoa como proximidade, consciência de localidade, geo-distribuição e organização hierárquica torna esse tipo de plataforma adequada para suportar restrições de energia em WSNs e WSANs. Além disso, essas são características típicas de computação em Névoa, e não de computação em Nuvem.

6.3.1.2. Análise de Dados

Enquanto os nós de Névoa fornecem uma computação localizada, permitindo assim baixa latência e consciência de contexto, a Nuvem oferece uma computação centralizada e global. A análise de dados é uma das aplicações fundamentais em Computação em Névoa, visto que muitas aplicações requerem tanto a localização da Névoa quanto a globalização da Nuvem, particularmente aquelas que fazem uso de análise de dados em tempo real.

Em [Bonomi et al. 2014] a aplicação da análise de dados em Névoa é iniciada a partir dos *gateways* na borda da rede, que coletam os dados gerados pelos sensores e dispositivos inteligentes. Alguns desses dados dizem respeito às aplicações que exigem processamento em tempo real como, por exemplo, em automação e controle de processos. A primeira camada da Névoa pode ser concebida para facilitar a interação M2M garantindo a coleta, o processamento dos dados e o controle dos dispositivos atuadores. Também pode filtrar os dados que serão processados localmente e enviar as informações resultantes para as camadas mais elevadas.

A segunda e a terceira camadas da Névoa podem permitir uma interação HMI tratando questões de visualização, relatórios, notificação, bem como sistemas e processos M2M. A escala de tempo dessas interações na Névoa podem variar de segundos a minutos para as análises em tempo real, até horas ou dias em análises transacionais. Como resultado, a Névoa tem de suportar vários tipos de armazenamento partindo de um modelo mais transitório na camada inferior até um suporte semi-permanente no nível mais elevado. É observado que quanto maior o nível da Névoa, maior a cobertura geográfica e mais longa a escala de tempo. A cobertura final global é fornecida pela Nuvem, que

é usada como repositório de dados com permanência de meses até anos, sendo tomada como base para análises voltadas à inteligência sobre negócios. Este é o ambiente HMI típico, com visualização dos principais indicadores de desempenho.

Como ilustrado na Figura 6.8, a medida que os dados são analisados na borda da rede e as informações são transferidas para o núcleo da Névoa elas podem ser usadas como retorno para aprimorar os modelos empregados, além de melhorar a tomada de decisão em tempo real. Diferentes métodos de otimização e predição podem ser usados como sistemas adaptativos, aprendizado de máquina, algoritmos genéticos, etc.

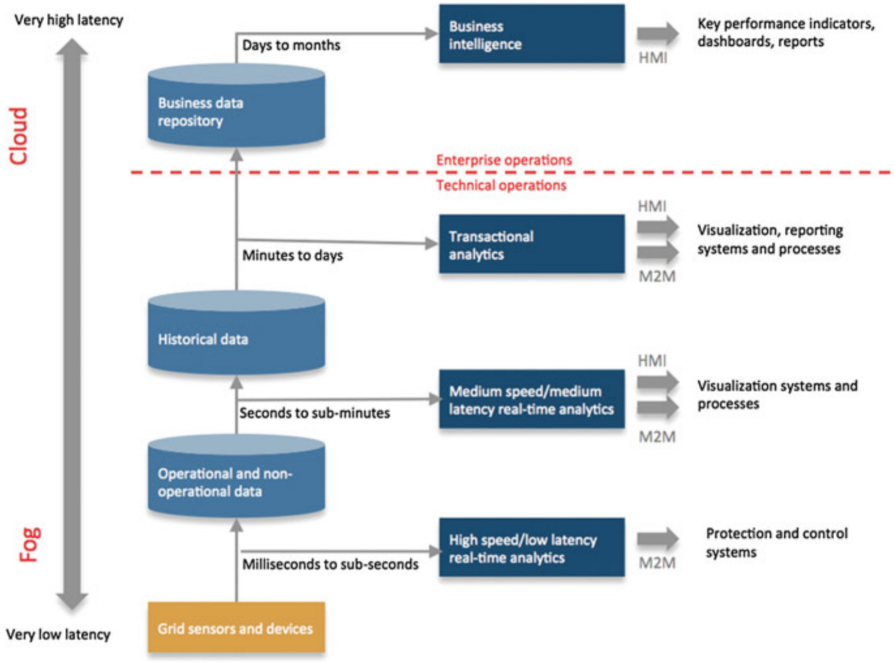


Figura 6.8. Análise de dados em Névoa: muitos usos para os mesmos dados. Fonte: [Bonomi et al. 2014]

6.3.1.3. Cache de Dados

Em [Peng 2004] *Content Delivery Network* (CDN) é definida como uma rede de armazenamento em *cache* implementada através de servidores na borda da Internet para reduzir o atraso de *download* de conteúdos a partir de locais remotos. Esse tipo de rede é projetada para servir usuários tradicionais, com interesses muito mais amplos e difíceis de prever do que usuários móveis. Atuando em uma área de serviço precisa, cada dispositivo de Névoa possui usuários mais definidos e uma demanda por serviços mais específicos.

Portanto, é fundamental para aplicações em Névoa explorar essas características buscando oferecer de forma plena sua capacidade de armazenamento e computação. De maneira similar, *Information Centric Network* (ICN) [Ahlgren et al. 2012] também define uma infraestrutura de *cache* sem fio que fornece serviços de distribuição de conteúdo para

usuários móveis. Mas de maneira diferente dos servidores *cache* em CDN e ICN, os dispositivos de Névoa são unidades de computação inteligentes.

Com isso, podem ser utilizados não apenas para armazenamento, mas também como uma infraestrutura de computação que interage com usuários e dispositivos móveis para o processamento de dados em tempo real. Os dispositivos de Névoa podem ser interconectados a Nuvem para utilizar um amplo poder de processamento e ferramentas de análise em *Big Data* em outras finalidades como descritas nesta seção.

Em [Bastug et al. 2014] é demostrado que os padrões de busca por informações de usuários móveis são previsíveis o suficiente para permitir que um sistema de *cache* obtenha essas informações antes que seus usuários as solicitem. Em uma área de serviço predefinida, um dispositivo de Névoa pode prevê o aumento da demanda dos usuários por certas informações e efetuar o armazenamento em *cache* dos dados mais acessados de forma proativa em sua memória local. Além disso, essas informações podem ser distribuídas geograficamente entre outros nós de Névoa com base nos locais específicos próximos ou mais distantes na hierarquia. Tais informações podem ser recuperadas a partir da Nuvem, ou acessadas pelo seu proprietário na borda da rede.

6.3.1.4. Gerenciamento de Dados sobre Saúde

Muitas vezes, a vida dos pacientes dependem de ações em um intervalo de tempo extremamente curto. No apóio a tomada de decisão rápida e eficiente, dispositivos inteligentes vêm ajudando os médicos tanto nas decisões quanto no monitoramento da saúde dos pacientes. O paradigma em Nuvem permanece importante nesta área, onde [Shi et al. 2015] discute as características da computação e dos serviços em Névoa que podem fornecer benefícios em sistemas voltados à saúde.

Para as soluções baseadas em Nuvem esta tem sido uma questão naturalmente sensível, pois os dados sobre a saúde contêm informações valiosas e privadas. Entretanto, a computação em Névoa permite que que os pacientes compartilhem e mantenham seus dados particulares de forma local e privada. Esses dados serão armazenados em nós de Névoa pessoais como celulares ou veículos inteligentes. O processamento desses dados (mas não os dados) será transferido de uma maneira autorizada para o dispositivo do paciente quando procurar ajuda de uma clínica médica ou um hospital. A alteração dos dados ocorre diretamente no dispositivo do paciente.

Em [Ahmad et al. 2016] é proposto um cenário sobre dados de saúde onde a computação em Névoa é usada como uma camada intermediária entre os usuários finais e a Nuvem para permitir maior controle e flexibilidade da privacidade dos clientes. Para isso, é introduzido um *Cloud Access Security Broker* (CASB) como componente integral da arquitetura, onde políticas de segurança podem ser aplicadas.

6.3.1.5. Segurança em Névoa

Não apenas na saúde, mas a segurança e integridade dos dados são duas características importantes e exigidas pela maioria das aplicações em IoT. Quanto mais tempo os dados

permanecem em "rota", mais vulneráveis eles se tornam para ataques, mesmo quando criptografados. Por isso, é sempre desejável ter pouco saltos entre clientes e servidores. A computação em Névoa pode oferecer a menor distância possível, proporcionando ainda vantagens da computação em Nuvem.

Mesmo sistemas de Nuvem localizados no interior da Internet podem sofrer ataques de disponibilidade através de métodos de Negação de Serviço (*Denial of Service*, DoS) [Sudha and Viswanatham 2013]. Esses tipos de ataques não precisam ser realizados diretamente sobre os próprios sistemas finais, uma vez que a desconfiguração dos dispositivos intermediários como roteadores podem ser igualmente fatais. Portanto, há muitas oportunidades para atingir sistemas de computação em Nuvem.

Por outro lado, os nós de Névoa são altamente distribuídos perto da borda da rede. Assim, para interferir completamente na disponibilidade destes sistemas é necessário um ataque em massa envolvendo todos os dispositivos que se encontram nas proximidades de um cliente. Isso exige métodos mais sofisticados e o emprego de mais recursos por parte dos atacantes do que em sistemas centralizados. Assim, é possível afirmar que de fora geral os sistemas de Computação em Névoa são menos propensos à ataques de negação de serviço do que os sistemas de computação em Nuvem.

A própria Névoa pode funcionar como um mecanismo de segurança, uma vez que outros mecanismos de proteção de dados existentes como criptografia podem falhar na prevenção de ataques, especialmente aqueles executados por invasores dentro de provedores. Em [Stolfo et al. 2012], é proposta uma abordagem diferente para a proteção de dados na Nuvem usando uma tecnologia ofensiva como armadilha (*decoy*). Nesta solução, o acesso aos dados na Nuvem é monitorado para detectar padrões de acesso anormais. Os acessos identificados como suspeitos são verificados através de perguntas contendo desafios. Caso não sejam respondidas corretamente, um ataque de desinformação é lançado sobre o invasor com o envio de grandes quantidades de informação *decoy*, protegendo os dados reais do usuário contra o acesso abusivo. Essa abordagem pode fornecer níveis sem precedentes de segurança para os dados dos usuários em um ambiente de Névoa.

6.3.1.6. Redes Inteligentes de Energia

Uma das mais avançadas aplicações de campo para Computação em Névoa reside nas Redes Inteligentes de Energia (*Smart Grids*), onde diferentes dispositivos como medidores e micro centrais de energia inteligentes são interconectadas através de uma rede de dados, como ilustrado na Figura 6.9. Neste contexto, aplicativos voltados ao balanceamento de carga de energia podem ser executados na borda da rede permitindo monitorar e controlar o consumo dos usuários ou alternar automaticamente entre energias alternativas de acordo com a demanda dos usuários, disponibilidade dos recursos e o menor preço.

Um algoritmo de gerenciamento de resposta à demanda centralizado em [Fadlullah et al. 2014] é implementado com uma abordagem de Computação em Nuvem, em que cada fornecedor e cliente se comunicam diretamente com a Nuvem. Porém, os algoritmos centralizados causam muito custo de largura de banda e gargalos, sendo inviáveis quando as redes reúnem um grande número de usuários [Jin et al. 2013]. Por outro lado, uma implementação totalmente distribuída onde cada usuário lida com seus próprios recursos e

Figura 6.9. Computação em Névoa em Redes Inteligentes de Energia. Fonte: [Stojmenovic 2014a]

controla seu consumo através da consulta frequente do preço dos fornecedores de energia requer uma computação em cada extremidade, além de uma comunicação extensiva pela distância entre fornecedores e clientes.

Em [Wei et al. 2014] é descrita uma solução intermediária através de uma rede inteligente de energia descentralizada e que emprega o paradigma em Névoa para fornecer aplicações essenciais. Recentemente, esta tecnologia foi considerada promissora na integração das redes de energia convencionais com fontes alternativas de energia como parques eólicos, células solares, micro-turbinas, etc. Nessa arquitetura, componentes chamados *macro-grids* e *micro-grids* possuem função equivalente aos dispositivos de Névoa na redução da sobrecarga de comunicação.

Na solução de gerenciamento de resposta à demanda em [Wei et al. 2014], cada um dos *macro-grids* ou *micro-grids* podem atuar como dispositivo de Névoa considerado diferentes camadas. Por exemplo, a primeira camada é formada pela interação com base nas informações locais e nos parâmetros fornecidos entre os consumidores conectados ao mesmo dispositivo de Névoa. A segunda camada é formada pela interação com base nas informações globais fornecidas pelos dispositivos de Névoa conectados ao mesmo servidor de Nuvem. Os clientes se comunicam com os dispositivos de Névoa nas proximidades em vez da Nuvem remota, e os dispositivos de Névoa frequentemente se comunicam com os clientes e, ocasionalmente, com a Nuvem.

Além disso, os dispositivos de Névoa podem ser interconectados, onde as informações parciais disponíveis nesses dispositivos permitem a formação de coalizões eficazes para minimizar as perdas de energia e reduzir o custo de comunicação. As *macro-stations* coordenam a transferência de energia entre os *micro-grids* e entre cada *macro-grid* e *macro-station*. Para reduzir de forma otimizada as perdas totais na rede elétrica, um algoritmo de formação de coalizões ganancioso é proposto em [Wei et al. 2014]. O algoritmo otimiza as perdas de potência totais em toda a rede de energia, incluindo o custo de carregamento e descarregamento nos dispositivos de armazenamento, e as perdas devido

às transferências de energia. O algoritmo cria intercambio entre pares de *micro-stations* dando prioridade aos pares com maior redução de perdas por unidade de energia trocada.

Outro exemplo de aplicação do conceito de Névoa está no problema do agendamento *on-line* da demanda por carregamento de energia em uma grande frota de veículos elétricos, onde o objetivo consiste em não sobrecarregar as redes de energia. A solução em [Jin et al. 2013] é baseada na classificação dinâmica de veículos elétricos heterogêneos em múltiplos grupos, onde uma abordagem de janela deslizante é utilizada no agendamento em tempo real da demanda de carga dentro de cada grupo. Neste processo, a informação sobre a carga atual de cada veículo é obtida *on-line* com a ajuda da Névoa.

6.3.1.7. Redes de Veículos Conectados

Em [Bonomi 2011] é descrito um cenário de veículos conectados a semáforos e postes de iluminação inteligentes, equipados com câmeras, sensores e pontos de acesso ilustrado na Figura 6.10. A comunicação de veículo para veículo (*Vehicle-to-Vehicle*, V2V), de veículos para pontos de acesso (*Vehicle-to-Infrastructure*, V2I) e diretamente entre os pontos de acesso enriquecem esse cenário. Os semáforos são capazes detectar as luzes piscando de uma ambulância e alterar automaticamente a sinalização das ruas para permitir a passagem do veículo de forma mais rápida pelo tráfego. Os postes de iluminação interagem localmente com sensores para detectar a presença de pedestres e ciclistas, medindo a distância e a velocidade dos veículos que se aproximam. Os semáforos podem sincronizar uma "onda verde"entre as avenidas e enviar sinais de alerta para evitar acidentes.

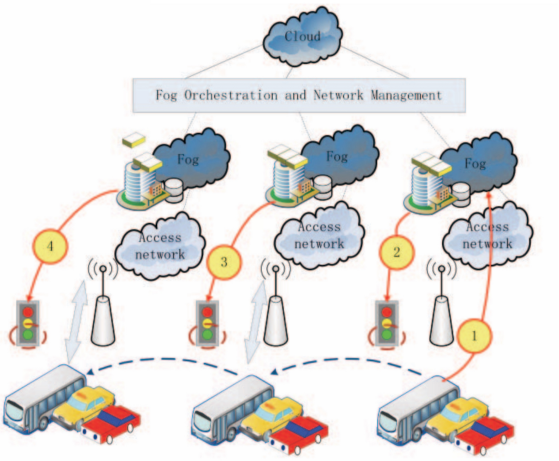


Figura 6.10. Cenário de veículos conectados. Fonte: [Stojmenovic 2014a]

A comunicação de dados em redes veiculares é usualmente realizada de forma descentralizada. Uma pesquisa mais extensa sobre métodos de encaminhamento e problemas em disseminação de dados nas redes veiculares *ad hoc* é apresentada em [Daraghmi et al. 2013]. Entretanto, o conceito de Névoa pode ser aplicado junto com SDN para tratar problemas fundamentais das redes veiculares como a conectividade intermitente, o grande número de colisões e alta taxa de perda de pacotes.

O conceito SDN em redes veiculares foi estudado e elaborado para o problema do *agendamento cooperativo de dados* em um ambiente de comunicação veicular híbrido descrito em [Liu et al. 2015]. No framework para redes veiculares proposto, os semáforos e as unidades inteligentes nas estradas funcionam como dispositivos de Névoa, assumindo a função dos roteadores no plano de dados SDN. O plano controle é implementado para monitorar e manter o estado individual dos veículos, otimizando o roteamento *multi-hop* nas redes V2V e V2I de uma forma logicamente centralizada.

A computação em Névoa pode fornecer um controle eficiente de semáforos em escala muito mais ampla do que as soluções existentes. Em [Zhou et al. 2011] um comando adaptativo de semáforos é empregado para maximizar o tráfego ao longo de pistas de mão única ou de mão dupla. Analisando o trajeto dos veículos, é possível diminuir o número de paradas e reduzir a emissão de gases do efeito estufa no ambiente. Em [Li and Shimamoto 2012] é proposta uma arquitetura em três camadas para calcular em tempo real a velocidade recomendada de veículos a partir da coleta de dados sobre o tráfego rodoviário utilizando: (*i*) dispositivos ETC (*Electronic Toll Collection*), (*ii*) antenas de rádio instaladas perto de semáforos e (*iii*) algoritmos para detecção de informações de tráfego com base em redes de sensores de proximidade [Zhang et al. 2013].

Uma solução baseada no paradigma em Névoa pode acomodar uma variedade de mecanismos para coleta de dados como etiquetas RFID em carros, sensores e câmeras de vídeo em semáforos, etc. As análises localizadas podem derivar informações de densidade de tráfego e fluxos em pontos específicos. Resumos dessas informações de tráfego devem ser encaminhados a partir dos dispositivos de Névoa para a Nuvem, possibilitando assim sua coordenação global. Mas os dispositivos de Névoa são tomadores de decisão e podem coordenar ações locais junto com seus vizinhos. Para suportar o controle centralizado, a disseminação de dados e a mobilidade entre os nós de Névoa é necessário que as informações sobre o estado dos veículos participantes sejam coletadas e migradas de forma eficiente. As informações sobre o estado dos veículos incluem sua localização em tempo real, velocidade, trajeto, capacidade de comunicação, etc.

6.3.1.8. Casas, Edifícios e Cidades Inteligentes

A aplicação da Internet das Coisas e Computação em Nuvem no desenvolvimento do conceito de Casas Inteligentes (*Smart Homes*) [Alam et al. 2012] tem aumentado nos últimos anos. Recentemente, grandes empresas têm oferecido seus primeiros dispositivos inteligentes e *gateways* voltados ao mercado pessoal. Entretanto, a sua adaptação ao cotidiano das casas é apenas o primeiro passo rumo a uma grande disseminação dessas tecnologias em outros ambientes ou cenários similares. Por exemplo, em [Nandyala and Kim 2016] é proposta uma arquitetura para o monitoramento de saúde usando as motivações e as vantagens da Computação em Névoa em casas inteligentes e hospitais.

Uma vez aprimoradas, estas tecnologias podem ser estendidas para ambientes mais amplos e complexos. Em [Stojmenovic 2014a] é descrito um cenário em Edifícios Inteligentes (*Smart Building*) onde o controle descentralizado pode ser facilitado por sensores sem fio implantados para fornecer a temperatura, a humidade, ou os níveis de vários gases na atmosfera do edifício. Além disso, os nós de Névoa podem trocar in-

formações com outros nós (por exemplo, no mesmo piso) para coordenar a combinação de leituras até alcançar medições confiáveis e realizar uma tomada de decisão distribuída para acionar outros dispositivos. Os componentes do sistema podem, então, trabalhar em conjunto para baixar a temperatura, injetar ar fresco, ou abrir janelas. Os condicionados de ar podem remover a umidade do ar ou aumentar a umidade. Sensores também pode rastrear e reagir a movimentos (por exemplo, ligando e desligando a luz). Os dispositivos de névoa distribuídos em cada andar podem colaborar em maior nível de atuação. Edifícios conectados podem manter a infraestrutura de seus ambientes externos e internos, para conservar energia, água e outros recursos.

Porém, a implantação de forma ubíqua de vários tipos de sensores em cenários futuristas e ainda mais abrangentes como em Cidades Inteligentes (*Smart Cities*) vai exigir uma complexa arquitetura em Névoa que suporte um grande número de componentes e serviços de infra-estrutura para Internet das Coisas. Em verdade, todos as aplicações descritas nessa seção além de muitas outras que não foram citadas ou que ainda serão criadas devem ser integradas, gerenciadas e compartilhadas de forma global, sobre diferentes infraestruturas de comunicação e provedores de Névoa. Por exemplo, [Tang et al. 2015] propõe uma arquitetura em Névoa em larga escala como forma de garantir a segurança de grandes comunidades. Nessa abordagem, a integração de diferentes redes de sensores geograficamente distribuídas e o suporte a análise de grandes volumes de dados é necessária para identificar eventos anômalos e perigosos, oferecendo respostas em tempo real.

6.3.2. Plataformas para Computação em Névoa

As plataformas em Névoa são desenvolvidas aproveitando sua proximidade às fontes de dados para suportar características tais como apoio à mobilidade, reconhecimento de localidade e baixa latência [Bonomi et al. 2012]. Suas arquiteturas devem incluir requisitos presentes nas arquiteturas em Nuvem como escalabilidade, virtualização, orquestração e multi-arrendatário. Na próxima seção, discutiremos os diferentes componentes que podem ser incorporados em sua organização hierárquica para garantir a distribuição dos recursos e serviços. As demais tópicos desta subseção apresentam exemplos de iniciativas que podem ajudar na implementação das aplicações apresentadas na Seção 6.3.1.

6.3.2.1. Arquiteturas de Plataformas em Névoa

Em [Yi et al. 2015a] é sugerida uma plataforma em névoa constituída pelos componentes ilustrados na Figura 6.11, onde alguns componentes são semelhantes aos da arquitetura de referência ilustrada na Figura 6.7 e descrita na Seção 6.2.5. Neste ambiente, as funções de Gerenciamento de Rede com bilhões de dispositivos heterogêneos executando diferentes serviços é uma tarefa desafiadora. Esses elementos distribuídos precisam ser configurados e coordenados, onde algumas tecnologias têm evoluído para domar a complexidade envolvida [Vaquero and Rodero-Merino 2014]:

- *Técnicas de Rede Definida por Software* (*network softwarization*) - como a Névoa pode abranger infraestruturas gerenciadas por diferentes organizações, é necessário que seus serviços sejam executados de forma homogênea ou idealmente automatizada por software. O uso de técnicas de virtualização baseadas em NFV pelas

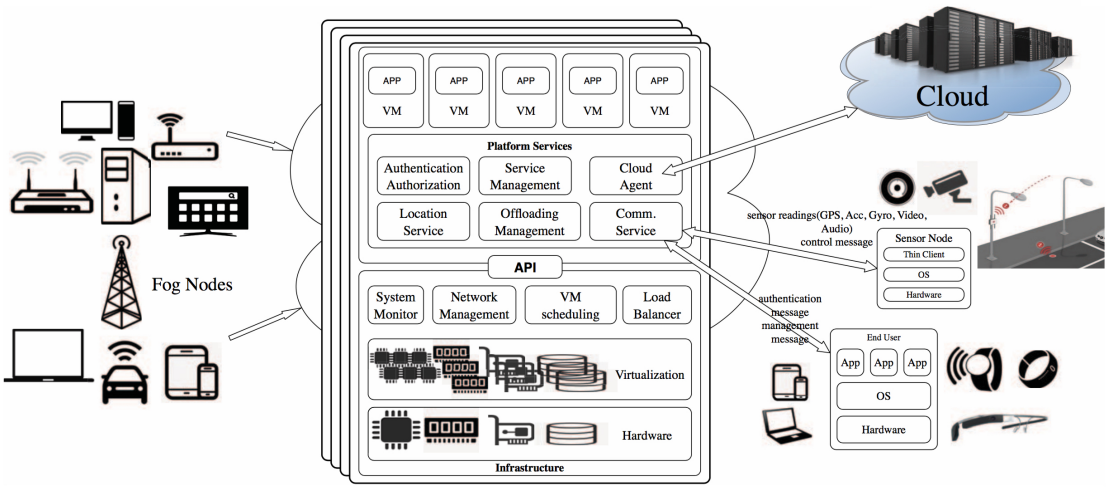


Figura 6.11. Componentes da Plataforma de Computação em Névoa. Fonte: [Yi et al. 2015a]

operadoras podem prover a implantação dinâmica de serviços sob demanda. O emprego de tecnologias baseadas em SDN permite que alguns serviços sejam estabelecidos apenas por software, resultando em operações mais ágeis e baratas do que soluções tradicionais baseadas em hardware. Por exemplo, a Figura 6.12 ilustra como os nós de Névoa podem ser implementados usando máquinas virtuais em uma nuvem local na borda da rede móvel LTE (*Long Term Evolution*) e como seu tráfego no núcleo EPC (*Evolved Packet Core*) pode ser rigidamente controlado graças à recursos de SDN.

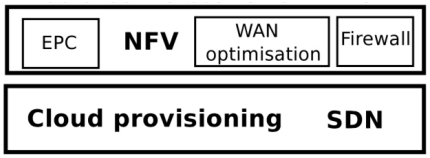


Figura 6.12. Uma Mini-nuvem na borda da rede móvel implementada com SDN como base para NFV. Fonte: [Vaquero and Rodero-Merino 2014]

- *Técnicas Declarativas e Assintóticas* - voltadas ao gerenciamento em larga-escala, estas soluções permitem a especificação do estado desejado para o sistema de forma declarativa ao invés do uso de comandos de configuração individuais. Entretanto, é assumido que o estado final almejado pode não ser alcançado uma vez que o sistema é passível de modificações durante o processo de configuração. Por exemplo, problemas de conexão podem levar a saída de nós da rede. A Cisco propõe uma nova abordagem para SDN utilizando técnicas de controle declarativo para lidar com a escala e a complexidade do gerenciamento em [OpFlex 2014].

- *Nós de Névoa* - um subconjunto de elementos e dispositivos na Névoa podem se comportar como mini-nuvens, onde duas diferentes implementações são mostradas na ilustração abaixo. Na Figura 6.13 (a), uma arquitetura *Cloudlet* em três camadas:

a camada inferior executa uma plataforma Linux e mantém uma *cache* de dados local da Nuvem, a camada central oferece virtualização usando um conjunto de softwares como [Openstack 2015], e a camada superior executa aplicações de forma isolada em instâncias diferentes de Máquinas Virtuais (VM). Outra abordagem na Figura 6.13 (b) é proposta em [Cisco IOx 2014], onde o software IOS no roteador torna-se parte de uma infra-estrutura SDN para habilitar recursos NFV e serviços de aplicação próximo dos clientes na borda da rede. Embora IOx seja uma plataforma proprietária, acompanha uma distribuição Linux e permite a instalação de outros sistemas operacionais, a execução de *scripts* e a compilação de código.

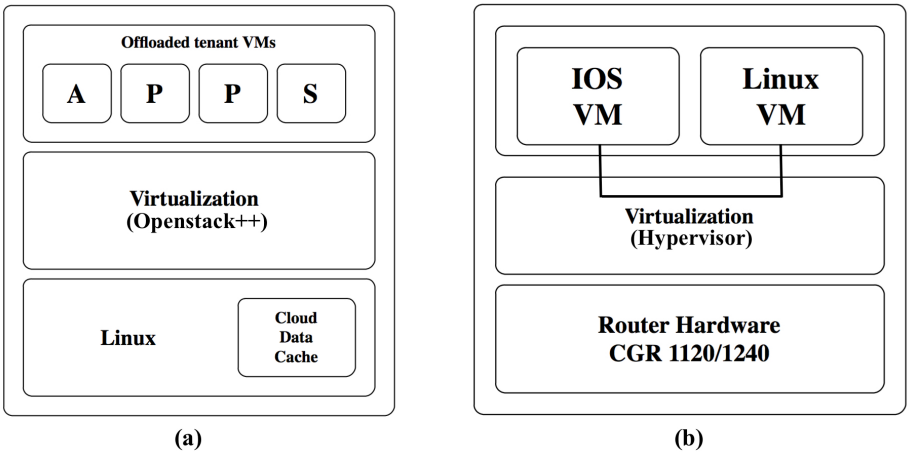


Figura 6.13. Arquitetura *Cloudlet* (a) e IOx (b) para nós de Névoa. Fonte: [Yi et al. 2015a]

- *Abordagens baseadas em Redes Peer-to-Peer (P2P) e de Sensores* - permitem uma cooperação entre os nós a fim de alcançar maior escalabilidade, porém com resultados similares às técnicas de gerenciamento que exigem um único provedor responsável pelo funcionamento da rede e dos serviços. Após anos de desenvolvimento e experimentação, as tecnologias P2P são suficientemente maduras para viabilizar a visão de Névoa explorando o conceito de localidade e eliminando a necessidade de um ponto de gerenciamento central. Recentemente, aplicações controversas como *Popcorn Time* [Idland et al. 2015] baseadas no protocolo *BitTorrent* [Shah and Pâris 2007] têm demostrado o potencial das redes P2P para fornecer serviços globais em larga escala. Muitas das ideias usadas em Redes de Distribuição de Conteúdo (*Content Distribution Network*, CDN) são aplicáveis também à Névoa através da troca de dados entre pequenas nuvens para reduzir o fluxo desnecessário e indesejado de informações para servidores em centros de dados distantes dos clientes.

O Monitor do Sistema é um elemento padrão em infraestruturas de Nuvem e fornece informações úteis para outros componentes. Por exemplo, pode ser empregado em funções de gerenciamento relacionadas com o descobrimento, a alocação, o provisionamento e a manutenção de um conjunto de recursos de forma distribuída. Particularmente, é usado pelo Balanceamento de Carga para distribuir trabalhos entre múltiplos nós de Névoa de modo a permitir a redundância e aumentar a disponibilidade dos serviços.

O Escalonamento de Máquinas Virtuais é responsável pela distribuição das instâncias de VMs com aplicações e recursos em uma sequência lógica considerando o uso do sistema, as estatísticas de carga de trabalho, as informações de localização e o modelo de mobilidade. Diferentes estratégias de programação são necessárias na tentativa de fornecer uma solução ideal para o escalonamento.

Por meio de uma API bem definida, esse conjunto de componentes descritos acima colaboram no compartilhamento virtualizado da infraestrutura de hardware (dispositivos, canais, nós, servidores, etc.) usando uma Arquitetura Baseada em Serviços (*Service-Oriented Architecture*, SOA). Essa camada fornece as funcionalidades necessárias à implementação dos Serviços de Plataforma através dos seguintes componentes:

- Gerenciamento de Serviços - oferece as principais funções que permitem a descoberta dinâmica, a monitoração e a configuração dos serviços. Esta componente possibilita a implantação remota de novos serviços em tempo de execução a fim de satisfazer as necessidades das aplicações. Um repositório pode ser construído para identificar o catálogo de serviços associados com cada objeto na rede para facilitar a composição de outros serviços mais complexos. Além disso, podem incluir outras funções relacionadas, por exemplo, com a Qualidade de Serviço (QoS) ou com questões semânticas como o gerenciamento de contexto.

- Serviços de Comunicação - devem manter a interoperabilidade entre os níveis da plataforma envolvendo dispositivos inteligentes com sensores e atuadores, nós de Névoa e a Nuvem. Em relação aos dispositivos, deve oferecer suporte a diferentes padrões de apresentação para garantir que os dados sejam propagados pelas aplicações em diferentes tipos de sistemas. Da mesma forma, os Agentes específicos devem garantir a comunicação com diferentes provedores de Nuvem, resguardando as particularidades na operação sobre cada modelo de serviço oferecido.

- Mecanismos de Autenticação e Autorização - localizados próximos dos usuários finais, a computação em Névoa abre uma porta para novos esquemas de autenticação e autorização através do uso de padrões de acesso, padrões de mobilidade e dispositivos de segurança confiáveis. Um trabalho relacionado em [Dsouza et al. 2014] propôs um esquema de controle de acesso de autorização de recursos heterogêneos.

- O Gerenciamento de *offloading* - tem impacto geral sobre a plataforma e se encarrega da transferência de tarefas a partir dos dispositivos para a Névoa. Em [Yi et al. 2015b] há uma pesquisa sobre a computação *offloading* em Névoa onde se destaca três problemas principais: (*i*) que informações são necessárias para decidir sobre *offloading*, (*ii*) como particionar uma aplicação para *offloading* e (*iii*) como projetar um esquema de *offloading* ideal, considerando que o custo do de transferência para névoa pode ser superior ao tempo de processando dos dados no próprio dispositivo.

- Serviços de Localização - precisam manter uma lista de localizações sobre os nós vizinhos (móveis ou não), rastrear os usuários móveis finais e compartilhar as informações de localização entre os nós de névoa envolvidos. Também realiza o mapeamento de locais de rede com locais físicos além da adoção um modelo de mobilidade que pode ser fornecido pelo usuário ou definido de forma autônoma.

O rastreamento e o mapeamento dos nós móveis precisará obter informações em diferentes níveis de comunicação como da camada física e de hardware (endereço físico, GPS, sensor IMU, etc.), da camada de rede (endereço IP) e da camada de aplicação (atividades sociais).

6.3.2.2. Plataforma OpenFog

Em novembro de 2015, a criação do primeiro consórcio voltado ao desenvolvimento de uma infraestrutura padrão para Computação em Névoa reuniu grandes empresas do mercado mundial como Cisco, Microsoft, Intel e ARM, junto com a Universidade de *Princenton* em *New Jersey*, EUA. O consórcio baseia-se na premissa de que uma arquitetura aberta é essencial para o sucesso de um ecossistema ubíquo de Computação em Névoa para plataformas e aplicações voltadas à Internet das Coisas.

A plataforma OpenFog é projetada como uma extensão do modelo em Nuvem tradicional, onde implementações em sua arquitetura podem residir em múltiplas camadas da topologia de rede. O objetivo é facilitar a implantação de aplicações com requisitos de interoperabilidade, desempenho, segurança, escalabilidade, capacidade de programação, confiabilidade, disponibilidade, facilidade de manutenção e agilidade.

Sua arquitetura deve oferecer suporte a uma infinidade de clientes ou dispositivos de borda. Ela pode funcionar em conjunto com serviços de Nuvem para realizar armazenamento, computação, comunicação em rede e tarefas de gerenciamento otimizadas com base em requisitos de carga de trabalho. Neste sentido, deve oferecer uma infra-estrutura necessária para permitir a construção de serviços *Fog-as-a-Service* (FaaS) visando tratar desafios em negócios. A infraestrutura e os componentes da arquitetura abaixo mostram como FaaS pode ser expandido sobre a arquitetura de referência.

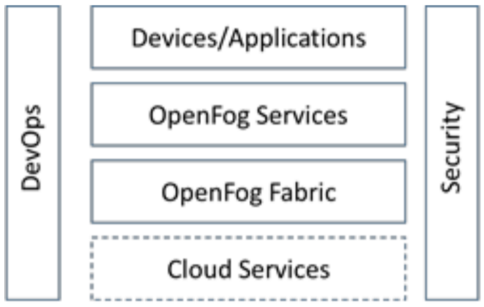


Figura 6.14. Visão da Infra-estrutura OpenFog. Fonte: [OpenFog 2016]

- *Serviços de Nuvem* - disponibilizam a infraestrutura em Nuvem para tarefas computacionais que precisam operar em um escopo mais amplo de informações ou sobre dados de borda pré-processados para estabelecer políticas. Estes devem ser usufruídos de forma a não impedir sua autonomia operacional.
- *Infraestrutura OpenFog* - é composto por componentes que permitem a construção de uma infraestrutura computacional homogênea em que serviços úteis podem ser

entregues aos ecossistemas agregados (por exemplo, dispositivos, *gateways* de protocolo e outros nós de Névoa). A infraestrutura homogênea é geralmente construída sobre hardware heterogêneo e plataformas desenvolvidas por vários fornecedores.

- *Serviços OpenFog* - são construídos sobre a infraestrutura OpenFog como uma arquitetura de micro-serviços em névoa. Estes serviços podem incluir aceleração de rede, NFV, SDN, entrega de conteúdo, gerenciamento de dispositivos, gerenciamento de topologia, processamento de eventos complexos, codificação de vídeo, *gateways* de protocolo, tráfego de *offloading*, *cache* de dados, criptografia, compressão, plataforma e algoritmos de análise, etc.

- *Dispositivos/Aplicações* - são sensores, atuadores e aplicativos em execução autônoma, dentro de uma infraestrutura de Névoa ou abrangendo diferentes provedores.

- *Segurança* - encapsulam as funcionalidades dentro de cada camada da arquitetura com mecanismos de controle de acesso para que a plataforma em Névoa e os ecossistemas agregados possam operar em um ambiente seguro, garantindo as transferências de dados entre seus componentes.

- *DevOps* - com foco em automação, são habilitados por um conjunto padronizado de procedimentos e *frameworks*. Fornecem agilidade para correções ou atualizações de software através de uma integração contínua e controlada.

Alguns temas recorrentes que aparecem no desenvolvimento de OpenFog representam os pilares da sua arquitetura. O emprego correto de cada um desses pilares como base para plataforma é a chave para uma implementação bem sucedida.

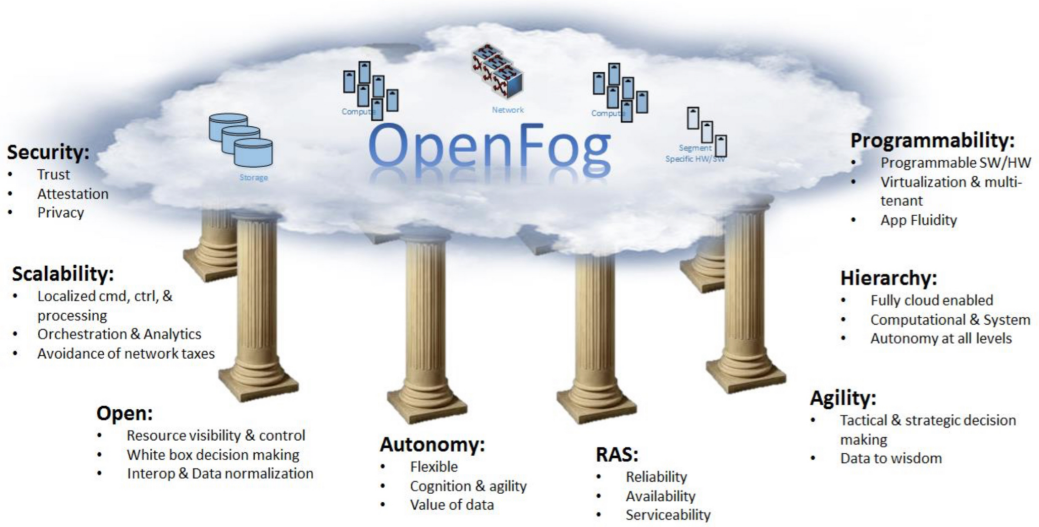


Figura 6.15. Pilares da plataforma OpenFog. Fonte: [OpenFog 2016]

6.4. Desafios para a Computação em Névoa

Neste tópico, vamos identificar e discutir potenciais desafios no contexto da Computação em Névoa, onde alguns deles podem indicar a direção para trabalhos futuros nesta área. Os desafios atuais serão introduzidos através da análise das soluções atuais e dos problemas que envolvem a integração entre os paradigmas de Nuvem e Névoa.

A fim de levantar os desafios relacionados a integração dessas tecnologias, será feito um paralelo entre os requisitos não atendidos pela computação em Nuvem, e como estes poderiam ser atendidos pela computação em Névoa. As redes de nevoeiro podem contornar os problemas para criar um ambiente mais flexível e confiável.

A seguir serão abordadas as questões envolvendo os modelos de programação e comunicação na Névoa, como os dispositivos envolvidos se comunicam e os problemas relacionados. A necessidade de gerenciamento de recursos e os principais desafios associados com a segurança, integridade e como prover privacidade nestes ambientes. Além das dificuldades em prover qualidade de serviço, implicando nas características de conectividade, confiabilidade, capacidade de processamento e armazenamento e atraso. Outros pontos a serem discutidos é um modelo de negócio sustentável, o impacto do consumo de energia e a necessidade de padronização para a Névoa.

6.4.1. Modelos de Programação

A transferência das tarefas de processamento para fora dos dispositivos ou *offloading* tem sido uma área de pesquisa ativa no domínio da computação móvel. Os dispositivos móveis que possuem restrições de recursos podem se beneficiar deste paradigma no desempenho de suas aplicações, com economia de armazenamento e tempo de bateria [Yi et al. 2015b]. No entanto, esta carga de trabalho é transferida para a Nuvem, que nem sempre é possível ou razoável de realizá-la devido às limitações e restrições que podem ser impostas [Dastjerdi et al. 2016].

Para tratar o problema do *offloading* [Orsini et al. 2015] propõem um *framework* adaptativo para Computação em Névoa chamado *CloudAware*, o qual destina tarefas de processamento tanto para a Nuvem quanto para os dispositivos localizados na borda da rede, ou seja, a Névoa. O *CloudAware* foi projetado com o objetivo de dar suporte às interações *ad-hoc* com baixa latência. Assim, visa facilitar o desenvolvimento de aplicações móveis e escaláveis, contribuindo para acelerar a computação com economia de energia e de largura de banda em diversos cenários de mobilidade dos usuários.

De acordo com [Yi et al. 2015b] o principal desafio de *offloading* na Computação em Névoa é como tratar com a dinamicidade do sistema. Pois, devemos levar em consideração que o acesso às redes *wireless*/radio, os nós e os recursos de Névoa são altamente dinâmicos. O processamento das tarefas fora dos dispositivos nesta infraestrutura está diante de novos desafios e oportunidades, tais como: *i*) definir a granularidade das tarefas a serem processadas, em diferentes hierarquias da Computação em Névoa e Computação em Nuvem; *ii*) particionar a aplicação dinamicamente para processamento das tarefas em Névoa e em Nuvem; e, *iii*) tomar decisões para adaptar o processamento das tarefas diante das mudanças acarretas pela dinamicidade da rede, dos dispositivos de Névoa e dos recursos, etc. Além disso, [Orsini et al. 2015] acrescentam a necessidade de replicação e

gerenciamento das aplicações em um ambiente não confiável e dinâmico.

É necessário unificar modelos de interfaceamento e programação para facilitar aos desenvolvedores a portabilidade de suas aplicações para Computação em Névoa. Isto se dá pelas seguintes razões: *i*) o modelo de computação centrado na aplicação será importante visto que permitirá otimizações para diferentes tipos de aplicação e seus componentes serão conscientes da aplicação; *ii*) é difícil para os desenvolvedores orquestrar recursos dinâmicos, hierárquicos e heterogêneos para construir aplicações compatíveis em diversas plataformas [Yi et al. 2015c].

[Hong et al. 2013a] apresenta uma API denominada *Mobile Fog* para o desenvolvimento de aplicações futurísticas que visa potencializar a larga escala, a distribuição geográfica e a garantia de baixa latência fornecida por uma infraestrutura em Névoa. A aplicação construída baseada na API *Mobile Fog* possui diversos componentes, cada um rodando em diferentes níveis na hierarquia dos dispositivos, semelhante a arquitetura de referência apresentada na seção 6.2.5. No entanto, ainda é necessário modelos mais gerais para diversas redes onde os nós de Névoa são nós móveis dinâmicos [Yi et al. 2015c].

6.4.2. Comunicação

Os aspectos relacionados à comunicação são desafiantes para os novos paradigmas de computação que tem surgido. É essencial para a sobrevivência dos sistemas e o funcionamento adequado das aplicações, principalmente, as que possuem sensibilidades a atrasos e requerem respostas em tempo real. Na Computação em Névoa, o dispositivo servidor é um componente intermediário da rede que se conecta com os usuários móveis, outros servidores de Névoa e a Nuvem. O servidor de Névoa pode monitorar o comportamento da rede e adaptar as aplicações de acordo com o ambiente [Luan et al. 2016].

[Suryawanshi and Mandlik 2015] e [Luan et al. 2016] afirmam que as pesquisas em comunicação devem levar em consideração como os componentes do sistema interagem. A seguir serão descritos os problemas associados e possíveis soluções envolvendo a comunicação, diante de três aspectos: *i*) dispositivos móveis e a Névoa; *ii*) a Névoa e a Nuvem; e, *iii*) entre as Névoas.

Comunicação entre os dispositivos móveis e a Névoa: ocorre diretamente entre os dispositivos móveis e os servidores de Névoa, através de conexões sem fio de único passo, tais como, WiFi, celular ou *bluetooth* [Luan et al. 2016]. Os servidores de Névoa possuem armazenamento limitado e entrega restrita de serviços localizados. Determinar os serviços de aplicação que ocasionam as menores taxas de faltas para os usuários móveis e selecionar os conteúdos para *cache* em cada servidor de Névoa devem ser temas de estudo. Para solucionar este problema é necessário prever os padrões de requisição de serviços dos usuários, a capacidade de armazenamento disponível e o poder de computação do servidor de Névoa. Como os servidores de Névoa são localizados em regiões específicas, os usuários apresentam características previsíveis na demanda de serviços. A Computação em Névoa também pode ser incorporada com a tecnologia de redes de celular 5G que está emergindo para prover maior cobertura e serviços dedicados.

Comunicação entre a Névoa e a Nuvem: se dá através de conexões sem fio e ca-

beada. A Nuvem é o depósito central de informações e o controlador central dos servidores de Névoa instalados em diferentes localizações. Os servidores de Névoa, localizados em diferentes regiões, podem selecionar os conteúdos da Nuvem e entregar cópias destas informações, a partir de sua *cache*, para os usuários móveis em localizações específicas. O servidor da Nuvem gerencia as aplicações e conteúdos para todo o sistema. Para isso, torna-se necessário tratar a confiabilidade e o controle escalável dos servidores de Névoa na Nuvem e desenvolver esquemas de roteamento de dados escaláveis. A entrega e atualização dos dados da Nuvem para a Névoa pode ser realizada através de SDN. Assim, o mecanismo de roteamento é separado entre o plano de controle e o plano de dados, onde a Nuvem gerencia a rede a partir de uma visão global para estabelecer os caminhos de roteamento dos dados e atualizar os servidores de Névoa distribuídos geograficamente.

Comunicação entre as Névoas: pode ser estabelecida através de conexões sem fio e cabeada. Os servidores de Névoa, localizados em distâncias próximas, podem pertencer a diferentes proprietários e oferecer serviços para usuários móveis similares. Os servidores de Névoa precisam colaborar para entrega de um serviço comum. A colaboração eficiente entre estes servidores, localizados em uma região próxima, pode aliviar o tráfego entre a Nuvem e a Névoa, melhorando o desempenho do sistema com economia de largura de banda e melhoria da taxa de dados. O roteamento entre os nós de Névoa pode ser realizado por dois modos: *i*) centralizado, através de uma abordagem baseada em SDN; e, *ii*) distribuído, através dos mecanismos tradicionais de roteamento, como por exemplo, OSPF (*Open Shorted Path First*).

A transmissão de dados entre os servidores de Névoa é desafiante pelas seguintes questões: *i*) *política de serviço*, os servidores de Névoa em diferentes localizações podem ser empregados por diversas entidades para usos comerciais distintos. Deste modo, podem apresentar heterogeneidade em suas políticas de serviços; *ii*) *topologia*, servidores de Névoa localizados em uma mesma região podem ser conectados à Internet através do mesmo Provedor de Serviços com alta taxa de conexão e baixo custo. Isso permite a colaboração eficiente entre os servidores próximos na Névoa buscando melhor desempenho do sistema, economia do custo associado à largura de banda e melhoria da taxa de dados. *iii*) *conexões*, o roteamento dos dados entre os servidores de Névoa precisam considerar as características destas conexões. Além de conexões cabeadas, podem ser estabelecidas conexões *wireless* oportunísticas, como por exemplo, em um sistema de veículos conectados em Névoa, onde os conteúdos são enviados entre os servidores de Névoa por contato veicular oportunístico [Luan et al. 2016].

6.4.3. Gerenciamento de Recursos

Geralmente, a Névoa é composta de dispositivos de diferentes tipos, os quais possuem capacidade de rede, poder de armazenamento e computação. Diante das suas especifidades, ainda é difícil para estes dispositivos corresponderem à capacidade de recursos dos tradicionais servidores, como ocorre na Computação em Nuvem.

A implantação de servidores de Névoa em diferentes localizações implica que o operador da rede precisa adaptar seus serviços. A aplicação instalada em cada servidor de Névoa precisa ser customizada baseada em sua demanda local. É necessário antecipar a

demanda de cada um dos servidores de Névoa a fim de empregar recursos de forma adequada, não prejudicando a escalabilidade do sistema. Frente às diferentes demandas de usuários em diferentes localizações, um grupo de servidores de Névoa podem colaborar para fornecer serviço aos usuários móveis próximos. No entanto, é preciso otimizar localmente o uso dos servidores de Névoa [Luan et al. 2016]. Deste modo, um gerenciamento de recurso eficiente é essencial para os ambientes de Computação em Névoa.

[Aazam and Huh 2015] apresenta um modelo de gerenciamento de recursos orientado a serviços que prevê o uso dos recursos pelos clientes e pré-aloca os recursos baseado no comportamento do usuário e a probabilidade de usá-los no futuro. Deste modo, a predição permite maior justiça e eficiência quando os recursos são consumidos. Em [Lewis et al. 2014] é proposto um mecanismo de provisionamento de recursos para *cloudlets* táticas, onde as aplicações são particionadas para executar de modo leve nos clientes, que rodam nos dispositivos móveis, e a computação intensiva é realizada nos servidores. Uma estratégia para fornecer infraestrutura para suportar computação *offloading* e plataforma de dados em dispositivos na borda da rede. *Cloudlets* se refere ao cenário em que o dispositivo de Névoa fornece infraestrutura para processar tarefas de outros dispositivos.

O gerenciamento de rede em Névoa é essencial para o oferecimento dos serviços de modo eficiente. Entretanto, oferecer serviços em cenários de larga escala, como ocorre em IoT, não é uma tarefa simples. [Yi et al. 2015b] sugere o uso de tecnologias que estão emergindo, como SDN e NFV. Estas tecnologias são propostas para criar ambientes de rede mais flexíveis e de fácil manutenção. O emprego de SDN e NFV na Computação em Névoa pode facilitar a implementação e o gerenciamento, aumentar a escalabilidade da rede e reduzir custos, tais como: alocação de recursos, migração de VM, monitoramento de tráfego, controle de aplicações cientes de contexto e interfaces programáveis.

6.4.4. Modelo de Negócio

A Computação na Névoa precisa de um modelo de negócio sustentável para manter-se funcionando de maneira adequada. De acordo com pesquisas realizadas, [Yi et al. 2015b] estabelece que os fornecedores de recursos da Névoa podem ser: *i*) fornecedores de serviços Internet ou suporte *wireless*, que podem construir a Névoa em suas infraestruturas; *ii*) fornecedores de serviço da Nuvem, que desejam expandir seu serviço centralizado da Nuvem para a borda da rede; *iii*) usuários fim, que desejam negociar sua computação extra, armazenamento de sua Nuvem privada local para reduzir os seus custos.

Entretanto, o modelo de negócio ainda não foi estabelecido, para isso é necessário resolver algumas questões, como por exemplo, em termos de faturamento, como se dará o preço dos diferentes recursos e qual será a fração de valor paga aos diferentes proprietários da Névoa. Para reforçar estas questões de política de preços torna necessário a contabilidade e o monitoramento da Névoa em diferentes granularidades, como ocorre no uso de serviços tradicionais.

Um modelo de negócio interessante para acelerar o uso da Computação em Névoa é o modelo baseado em incentivos aos usuários. Os proprietários de Nuvens privadas locais, localizadas na borda da rede, podem fornecer seus serviços para a Névoa, com capacidade de computação e armazenamento. A partir da perspectiva técnica de Computação na Nuvem e virtualização, o fornecedor de serviço para Névoa pode arrendar o seu

poder de computação e armazenamento ocioso e receber pagamento para reduzir os seus próprios custos [Yi et al. 2015b].

6.4.5. Segurança, Integridade e Privacidade

A Computação em Nuvem enfrenta problemas de segurança que podem ser estendidos para Névoa. Além disso, são adicionados desafios de segurança ao ambiente de Computação em Névoa por sua localização e natureza descentralizada, instalados em locais sem proteção e o devido rigor na vigilância [Dastjerdi et al. 2016]. Os ataques empregados na Computação em Névoa podem comprometer a disponibilidade do sistema e usuários maliciosos podem praticar espionagem com sequestro de dados [Stojmenovic et al. 2015].

Detecção de intrusos. Os mecanismos usados para proteção de dados, como a encriptação, tem falhado em prevenir ataques de roubos de dados, principalmente quando executados internamente no provedor de serviço da Nuvem [Stolfo et al. 2012]. Considerando a Névoa como uma pequena Nuvem, podemos aplicar técnicas de detecção de intrusos [Modi et al. 2013]. Segundo [Stojmenovic et al. 2015], a intrusão nestes ambientes pode ser detectada através do uso de método baseado em assinatura e método baseado em anomalias. No método baseado em assinatura, os padrões de comportamento do usuário são observados e checados com um banco de dados existente de possíveis maus comportamentos. No método baseado em anomalias, o comportamento observado é comparado com o comportamento esperado para verificar se há desvios.

Em [Stolfo et al. 2012], os autores propõem uma abordagem para prover segurança aos dados na Nuvem através da análise do perfil de comportamento do usuário e o uso ofensivo da tecnologia *decoy*. Os dados são monitorados continuamente para detectar padrões de acesso anormais, decorrentes do perfil de comportamento do usuário. Quando um acesso é considerado suspeito, constatado através de perguntas de desafio, é lançado um ataque de desinformação, retornando grandes quantidades de informações *decoy* para o atacante. Assim, protege os dados do usuário do uso indevido. O ambiente de Névoa é utilizado para disparar armadilhas, arquivos *decoy*, quando um ataque é detectado. Deste modo, não causa interferência nas atividades normais dos usuários.

[Stojmenovic et al. 2015] expuseram os problemas de segurança em Névoa a partir de estudos sobre um típico ataque de homem do meio. Nos experimentos realizados, os *gateways* que servem como dispositivos de Névoa foram comprometidos. Os usuários da Névoa se conectam a pontos de acesso falsos que fornecem serviços enganosos como legítimos. O cenário estabelecido é uma comunicação por vídeo chamada, no qual o usuário utiliza a tecnologia 3G para enviar dados para um usuário na WLAN.

Uma vez que o atacante toma o controle do *gateway*, a comunicação privada das vítimas pode ser sequestrada. O atacante pode retransmitir ou modificar os dados em seu próprio computador e enviá-los para o *gateway*. Assim, o *gateway* transmite os dados do atacante para o usuário na WLAN. O experimento teve como objetivo avaliar as características deste ataque através da análise do consumo de CPU e memória. A partir dos experimentos, concluiu-se que o ataque homem do meio em Névoa pode não ser identificado, visto que o consumo de memória e CPU é desprezível. Os autores concluem que o ataque homem do meio é difícil de evitar e defender, tendo potencial para se tornar um típico ataque na Computação em Névoa.

Modelo de Confiança. A Névoa pode ter diferentes provedores, o que dependerá do modelo de negócio implementado, conforme descrito na seção 6.4.4. No entanto, a flexibilidade existente compromete a confiança da Névoa. Um nó de Névoa pode agir como trapaceiro e influenciar outros nós a se conectarem a ele. Uma vez conectados, os nós trapaceiros podem manipular as requisições dos usuários fins ou da Nuvem, coletar ou até mesmo adulterar os dados para lançar novos ataques. Modelos de confiança baseados em reputação tem sido amplamente empregados em P2P, comércio eletrônico e redes sociais online. Segundo [Yi et al. 2015c], o projeto de um sistema de reputação para Computação em Névoa precisa levar em consideração as seguintes questões: *i*) como alcançar identidades persistentes, únicas e distintas; *ii*) como tratar ataques intencionais e acidentais; e, *iii*) como punir e resgatar a reputação dos nós. A existência de nós trapaceiros será uma grande ameaça para a segurança e privacidade dos dados. Este problema é difícil para tratar em ambientes de Névoa por conta dos diferentes esquemas de gerenciamento de confiança e da dinamicidade do sistema, que torna difícil a manutenção de uma lista com os nós desonestos.

Autenticação. [Stojmenovic and Wen 2014] consideram que o principal desafio da segurança na Computação em Névoa é a autenticação, em vários níveis dos nós. O uso de Infraestrutura de Chave Pública (PKI - *Public Key Infrastructure*) pode solucionar este problema [Chen et al. 2014]. No entanto, [Yi et al. 2015c] afirmam que as autenticações baseadas nas tradicionais PKI não são eficientes e não são escaláveis. Em [Stojmenovic et al. 2015] é proposto um tipo de autenticação chamada *Stand-Alone Authentication* que é capaz de autenticar o usuário mesmo quando não há conexão com o servidor da Nuvem. O direito de autenticação é delegado pelo Servidor de Autenticação para um dispositivo de Névoa. Esta abordagem é baseada em uma encriptação híbrida, no uso do Padrão de Criptografia Avançado (AES - *Advanced Encryption Standard*) e de *smart card*.

Técnicas de execução em ambiente confiáveis (*Trusted Executed Environment*, TEE) são propostas como uma potencial solução para os problemas relacionados à autenticação em Computação na Névoa [Marforio et al. 2014]. Métodos baseado em cálculo de influência podem ser usados para detectar nós de Névoa trapaceiros ou desqualificados e, assim, reduzir o custo com a autenticação [Han et al. 2011] [Behrisch et al. 2011]. As tecnologias emergentes de autenticação baseadas em biometria para dispositivos móveis e Nuvem irão beneficiar a Computação em Névoa [Yi et al. 2015c].

Controle de Acesso. O controle de acesso tem sido uma ferramenta confiável para garantir a segurança em sistemas que envolvem dispositivos inteligentes e Nuvem [Yi et al. 2015b]. Em virtude da natureza da Computação em Nuvem estar relacionada com a prestação de serviços, o controle de acesso é usualmente implementado através de criptografia [Yi et al. 2015c]. [Yu et al. 2010] apresenta um controle de acesso de dados baseado em fina granularidade, através da exploração de técnicas de encriptação baseadas em atributos (*Attribute-Based Encryption*, ABE). Em [Dsouza et al. 2014] é proposto um controle de acesso baseado em políticas para Computação em Névoa. Neste trabalho, foram identificados os desafios existentes em políticas de gerenciamento para Computação em Névoa que tornam críticos o suporte ao compartilhamento seguro, a colaboração e reuso de dados em ambientes heterogêneos. O *framework* apresentado visa o gerenciamento de políticas acompanhado com critérios de políticas e esquemas relevantes. Os autores demonstram a viabilidade e aplicabilidade da proposta através de cenários de caso

de uso em sistemas de transporte inteligente que requer análise e agregação de dados em tempo real e transmissão dinâmica de informações entre os dispositivos que fazem parte dos sistemas. Durante a colaboração e compartilhamento de dados podem surgir conflitos e problemas, os quais podem ser tratados dinamicamente pelo *framework* enquanto ocorre a transferência de informações de forma segura para o destino final.

Integridade. Diversas aplicações na Internet demandam por segurança e integridade [Zao et al. 2014]. Se os dados permanecem em rota por mais tempo, mais vulneráveis eles estão aos ataques, mesmo quando encriptados. Assim, sempre é desejável ter poucos passos entre os clientes e os servidores. A Computação em Névoa fornece a menor distância possível agregando outras vantagens da Nuvem. Deste modo, a Computação em Névoa é preferível à Computação em Nuvem em muitas situações [Firdhous et al. 2014].

Privacidade. De modo geral, os usuários da Internet estão preocupados com o risco da falta de privacidade. Mecanismos que preservam a privacidade tem sido proposto em diversos cenários, como por exemplo, Nuvem, redes sem fio, redes sociais, entre outros. A preocupação dos usuários não é diferente nos ambientes de Computação em Névoa, principalmente, por conta da natureza deste ambientes, onde os dados dos usuários podem ser processados em *hardwares* e *softwares* de outros proprietários. Com isto, introduz preocupações sobre a privacidade dos dados e sua visibilidade por partes não autorizadas. Por esta razão precisamos investigar técnicas e mecanismos para assegurar a confiança entre as partes que estão cooperando [Suryawanshi and Mandlik 2015]. Na Névoa, os algoritmos utilizados para preservar a privacidade podem ser executados entre a Névoa e a Nuvem, desde que os recursos de computação e armazenamento sejam suficientes para ambos, o que nem sempre é possível nos dispositivos fins. No entanto, técnicas como encriptação homomórfica podem ser usadas para permitir privacidade durante a agregação dos dados nos *gateways* locais sem decriptação [Yi et al. 2015b].

6.4.6. Qualidade de Serviço

A qualidade de serviço é uma métrica importante para o provimento de serviços através da Névoa. [Yi et al. 2015b] analisa a qualidade de serviço da Computação em Névoa sob quatro aspectos: conectividade, confiabilidade, capacidade e atraso. Cada um destes aspectos serão descritos a seguir:

Conectividade. O ambiente de Névoa envolve dispositivos heterogêneos com capacidade de expandir a conectividade da rede. A retransmissão na rede, particionamento e agrupamento fornece novas oportunidades para reduzir custos. A seleção de nós de Névoa pelos usuários terá um grande impacto no desempenho do sistema. Para otimizar o desempenho e disponibilidade dos serviços de Névoa, pode-se selecionar dinamicamente um subconjunto de nós de Névoa como retransmitidores para uma determinada área ou usuário, com restrições de atrasos, largura de banda, conectividade e consumo de energia.

Confiabilidade. É uma das primeiras preocupações quando projetamos sistemas de Computação em Névoa, onde há integração de um grande número de dispositivos distribuídos geograficamente. Segundo [Madsen et al. 2013], para termos uma Névoa confiável é essencial levarmos em consideração as falhas que podem ocorrer, tais como: *i*) os dispositivos podem falhar individualmente; *ii*) falhas na rede e falta de cobertura de rede em algumas regiões; *iii*) falhas na plataforma de serviço; e, *iv*) falhas na interface do

usuário conectado ao sistema. Normalmente, a confiabilidade pode ser melhorada através de *check-pointing* periódicos para recuperação após as falhas ocorrerem, reescalonamento das tarefas falhas ou replicação para explorar as execuções em paralelo. No entanto, em um ambiente altamente dinâmico como uma Névoa, pode não ser possível a recuperação e o reescalonamento. Caso houvesse, introduziria latência e não poderia adaptar-se às mudanças. A replicação pode ser usada em vários nós de Névoa, que devem cooperar para melhorar o desempenho do sistema.

Capacidade. Possui dois aspectos: largura de banda e capacidade de armazenamento. Para alcançar altas taxas de largura de banda e armazenamento deve-se analisar como os dados são distribuídos nos nós de Névoa, considerando a sua localização. Trabalhos foram desenvolvidos na área de Nuvem e de redes de sensores. Estes problemas estão diante de novos desafios na Computação em Névoa que vêm desde o projeto, na interação entre a Névoa e a Nuvem e em como acomodar diferentes cargas de trabalho. Além dos problemas relacionados a busca de conteúdos dispersos nos nós de Névoa devido a dinâmica da localização dos dados e grande capacidade total de volume. A economia de largura de banda e redução de atrasos pode ser obtida pelo uso de *cache* nos nós de Névoa, sendo interessante reprojetar a *cache* para explorar localidade temporal e ampla cobertura no ambiente de Névoa.

Atraso. Algumas aplicações precisam da Computação na Névoa para fornecer processamento de *streaming* em tempo real. Estas aplicações são sensíveis à latência. Para satisfazer as exigências de latência das aplicações, [Hong et al. 2013b] propõe um sistema de processamento de evento espaço temporal oportunístico que usa tratamento de requisições continuamente baseado em predição. O sistema prediz futuras requisições em uma região para usuários em movimento e antecipam o processamento de eventos para tornar as informações disponíveis quando os usuários alcançarem a localização. Após avaliação, o sistema proposto alcançou resultados significativos, com latência próxima de zero em vários casos. Em [Ottenwälder et al. 2014] é apresentado o RECEP, um sistema para aumentar a escalabilidade dos sistemas móveis que exigem o processamento de eventos complexos (*Complex Event Processing*, CEP) e a agregação de dados de sistemas distribuídos em tempo real. O RECEP explora a sobreposição de interesses dos usuários móveis através de métodos que utilizam o processamento de maneira eficiente para reduzir a requisição de recursos.

6.4.7. Consumo de energia

Os ambientes de Névoa são formados por um grande número de dispositivos distribuídos geograficamente, tendo a computação distribuída como principal aspecto. Levando em consideração a natureza dos ambientes em Névoa, estes podem ser menos eficentes em energia do que os ambientes centralizados em Nuvem. Deste modo, a redução do consumo de energia em ambientes de Névoa se torna um desafio [Dastjerdi et al. 2016].

Em [Deng et al. 2015] é apresentado um estudo sobre o *trade-off* entre o poder de consumo e o atraso em sistemas de Computação em Núvem e Névoa. Os autores formalizaram matematicamente o problema de alocação da carga de trabalho entre a Névoa e a Nuvem. Decomporam o problema principal em três subproblemas: *i*) *trade-off* entre o poder de consumo e o atraso em Computação na Névoa; *ii*) *trade-off* entre o poder de con-

sumo e o atraso em Computação na Nuvem; e, *iii*) minimização do atraso de comunicação no envio, através do subsistema WAN (*Wide Area Network*), durante a transmissão dos dados da Névoa para Nuvem. Através de simulações e resultados numéricos foi possível mostrar que a Computação em Névoa pode melhorar significativamente o desempenho da Nuvem através da economia de largura de banda e redução na latência de comunicação. Os autores concluem que os subproblemas podem ser solucionados independentemente através de técnicas de otimização, dentro do seu subsistema correspondente.

6.4.8. Padronização

Os mecanismos de padronização se tornam necessários para que cada membro da rede (terminal, dispositivos na borda da rede, etc.) possa interagir e cooperar. Os protocolos são necessários para que os membros da rede anunciem sua disponibilidade para hospedar componentes de *software* de terceiros e para que outros usuários possam enviar suas tarefas para serem executadas. Deste modo, é fundamental o desenvolvimento de padrões de protocolos, arquiteturas e APIs para facilitar a interconexão entre objetos inteligentes heterogêneos e a criação de serviços aprimorados que possam satisfazer as necessidades dos usuários [Suryawanshi and Mandlik 2015].

6.5. Considerações Finais

A Computação em Névoa é construída pela convergência de um conjunto de tecnologias que atravessaram processos de desenvolvimento e amadurecimento independentes. A integração destas tecnologias em um cenário único busca responder às novas exigências introduzidas pela ubiquidade dos dispositivos, requerendo mais agilidade das redes de comunicação e do gerenciamento de serviços em Nuvem, além de mecanismos extensíveis para privacidade dos dados.

De forma geral, uma infraestrutura em Névoa oferece às tecnologias de Nuvem mecanismos para lidar com o imenso volume de dados gerados diariamente pela Internet das Coisas. O processamento mais próximo de onde os dados são produzidos é a forma mais adequada de resolver os desafios atuais relacionados a sua explosão em volume, variedade e velocidade. As tecnologias de Névoa também podem acelerar a consciência e a resposta à eventos, eliminando requisições e transferências custosas de dados da borda da rede para o centro da Nuvem. Também protege informações sensíveis e valiosas extraídas da Internet das Coisas, analisando os dados dentro dos muros das empresas e organizações. Como resultado, a Névoa vai mudar dramaticamente muitas das práticas atuais em quase todas as camadas das arquiteturas de Nuvem, no suporte ao desenvolvimento de aplicações, no gerenciamento de recursos e serviços, contabilidade, colaboração, etc.

A Computação em Névoa vai dar origem a novas formas de competição e cooperação entre os provedores na Internet. Entretanto, não é fácil determinar como os diferentes atores no mercado irão se alinhar para oferecer serviços em Névoa de forma global nos próximos anos. É previsto que novos protagonistas entrarão em cena no papel de usuários ou provedores. As organizações que adotarem a Computação em Névoa devem obter uma percepção mais profunda e rápida das informações, levando a um aumento na agilidade dos negócios para alcançar níveis de serviço e segurança mais elevados.

Referências

[Aazam and Huh 2015] Aazam, M. and Huh, E.-N. (2015). Dynamic Resource Provisioning Through Fog Micro Datacenter. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*, pages 105–110. IEEE.

[Aazam et al. 2014] Aazam, M., Khan, I., Alsaffar, A. A., and Huh, E.-N. (2014). Cloud of Things: Integrating Internet of Things and Cloud Computing and The Issues Involved. In *Applied Sciences and Technology (IBCAST), 2014 11th International Bhurban Conference on*, pages 414–419. IEEE.

[ABI Research 2015] ABI Research (2015). Internet of Everything Semiannual Update. Disponível em: https://www.abiresearch.com/market-research/service/internet-of-everything/. Acesso em: 16 dez. 2015.

[Ahlgren et al. 2012] Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., and Ohlman, B. (2012). A Survey of Information-Centric Networking. *Communications Magazine, IEEE*, 50(7):26–36.

[Ahmad et al. 2016] Ahmad, M., Amin, M. B., Hussain, S., Kang, B. H., Cheong, T., and Lee, S. (2016). Health fog: a novel framework for health and wellness applications. *The Journal of Supercomputing*, pages 1–19.

[Alam et al. 2012] Alam, M. R., Reaz, M. B. I., and Ali, M. A. M. (2012). A review of smart homes - past, present, and future. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(6):1190–1203.

[Atzori et al. 2010] Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of Things: A Survey. *Computer Networks*, 54(15):2787–2805.

[Auto-ID 2016] Auto-ID (2016). Auto-ID Labs at MIT. Disponível em: http://www.autoidlabs.org/. Acesso em: 20 fev. 2016.

[Bastug et al. 2014] Bastug, E., Bennis, M., and Debbah, M. (2014). Living on the edge: The role of proactive caching in 5g wireless networks. *Communications Magazine, IEEE*, 52(8):82–89.

[Behrisch et al. 2011] Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D. (2011). Sumo–Simulation of Urban Mobility. In *The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain*.

[Bonomi 2011] Bonomi, F. (2011). Connected Vehicles, The Internet of Things, and Fog Computing. In *The Eighth ACM International Workshop on Vehicular Inter-Networking (VANET), Las Vegas, USA*, pages 13–15.

[Bonomi et al. 2014] Bonomi, F., Milito, R., Natarajan, P., and Zhu, J. (2014). Fog Computing: A Platform for Internet of Things and Analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*, pages 169–186. Springer.

[Bonomi et al. 2012] Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog Computing and Its Role in the Internet of Things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

[Botta et al. 2016] Botta, A., de Donato, W., Persico, V., and Pescapé, A. (2016). Integration of Cloud Computing and Internet of Things: A Survey. *Future Generation Computer Systems*, 56:684–700.

[Chen et al. 2014] Chen, C., Raj, H., Saroiu, S., and Wolman, A. (2014). cTPM: a Cloud TPM for Cross-Device Trusted Applications. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*.

[Christophe et al. 2011] Christophe, B., Boussard, M., Lu, M., Pastor, A., and Toubiana, V. (2011). The web of things vision: Things as a service and interaction patterns. *Bell labs technical journal*, 16(1):55–61.

[Cisco IOx 2014] Cisco IOx (2014). Cisco IOx Technical-Overview. Disponível em: https://developer.cisco.com/site/iox/technical-overview/. Acesso em: 22 mar. 2016.

[Daraghmi et al. 2013] Daraghmi, Y.-A., Yi, C.-W., and Stojmenovic, I. (2013). Forwarding methods in data dissemination and routing protocols for vehicular ad hoc networks. *Network, IEEE*, 27(6):74–79.

[Dash et al. 2010] Dash, S. K., Mohapatra, S., and Pattnaik, P. K. (2010). A Survey on Applications of Wireless Sensor Network Using Cloud Computing. *International Journal of Computer science & Engineering Technologies (E-ISSN: 2044-6004)*, 1(4):50–55.

[Dastjerdi et al. 2016] Dastjerdi, A. V., Gupta, H., Calheiros, R. N., Ghosh, S. K., and Buyya, R. (2016). Fog Computing: Principals, Architectures, and Applications. *arXiv preprint arXiv:1601.02752*.

[Deng et al. 2015] Deng, R., Lu, R., Lai, C., and Luan, T. H. (2015). Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing. In *2015 IEEE International Conference on Communications (ICC)*, pages 3909–3914.

[Díaz et al. 2016] Díaz, M., Martín, C., and Rubio, B. (2016). State-of-the-art, Challenges, and Open Issues in the Integration of Internet of Things and Cloud Computing. *Journal of Network and Computer Applications*.

[Dinh et al. 2013] Dinh, H. T., Lee, C., Niyato, D., and Wang, P. (2013). A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches. *Wireless communications and mobile computing*, 13(18):1587–1611.

[Distefano et al. 2012] Distefano, S., Merlino, G., and Puliafito, A. (2012). Enabling the Cloud of Things. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 858–863. IEEE.

[Dsouza et al. 2014] Dsouza, C., Ahn, G.-J., and Taguinod, M. (2014). Policy-driven Security Management for Fog Computing: Preliminary Framework and a Case Study. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*, pages 16–23. IEEE.

[Duan et al. 2015] Duan, Y., Fu, G., Zhou, N., Sun, X., Narendra, N. C., and Hu, B. (2015). Everything as a Service (XaaS) on the Cloud: Origins, Current and Future Trends. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pages 621–628. IEEE.

[Elijah 2016] Elijah, G. (2016). Carnegie Mellon University - Elijah Gabriel Reseach Group: Cloudlet-based Mobile Computing. Disponível em: http://elijah.cs.cmu.edu. Acesso em: 22 mar. 2016.

[ETSI ISG MEC 2015] ETSI ISG MEC (2015). ETSI Industry Specification Group, Mobile Edge Computing. Disponível em: http://www.etsi.org/technologies-clusters/technologies/mobile-edge-computing. Acesso em: 10 jan. 2016.

[Fadlullah et al. 2014] Fadlullah, Z. M., Quan, D. M., Kato, N., and Stojmenovic, I. (2014). Gtes: An optimized game-theoretic demand-side management scheme for smart grid. *Systems Journal, IEEE*, 8(2):588–597.

[Firdhous et al. 2014] Firdhous, M., Ghazali, O., and Hassan, S. (2014). Fog Computing: Will it be the Future of Cloud Computing? In *Proceedings of the 3rd International Conference on Informatics & Applications, Kuala Terengganu, Malaysia*, pages 8–15.

[Fox et al. 2012] Fox, G. C., Kamburugamuve, S., and Hartman, R. D. (2012). Architecture and Measured Characteristics of a Cloud Based Internet of Things. In *Collaboration Technologies and Systems (CTS), 2012 International Conference on*, pages 6–12. IEEE.

[Fritsch and Walker 2014] Fritsch, J. and Walker, C. (2014). The Problem with Data. In *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, pages 708–713. IEEE Computer Society.

[Gao et al. 2015] Gao, Y., Hu, W., Ha, K., Amos, B., Pillai, P., and Satyanarayanan, M. (2015). Are cloudlets necessary? Disponível em: http://reports-archive.adm.cs.cmu.edu/anon/anon/2015/CMU-CS-15-139.pdf. Acesso em: 15 mar. 2016.

[Gartner Inc. 2015] Gartner Inc. (2015). Gartner's 2015 Hype Cycle for Emerging Technologies Identifies the Computing Innovations That Organizations Should Monitor. Disponível em: http://www.gartner.com/newsroom/id/3114217. Acesso em: 20 fev. 2016.

[Gartner Inc. 2016] Gartner Inc. (2016). Research Methodologies: Gartner Hype Cycle. Disponível em: http://www.gartner.com/technology/research/methodologies/hype-cycle.jsp. Acesso em: 15 fev. 2016.

[Github 2016] Github (2016). Elijah Gabriel Research Group - Open Edge Computing. Disponível em: https://github.com/openedgecomputing. Acesso em: 22 mar. 2016.

[Google Trends 2016] Google Trends (2016). Google Trends. Disponível em: http://www.google.com/trends. Acesso em: 11 fev. 2016.

[Greenberg et al. 2008] Greenberg, A., Hamilton, J., Maltz, D. A., and Patel, P. (2008). The cost of a cloud: Research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73.

[Gubbi et al. 2013] Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Generation Computer Systems*, 29(7):1645–1660.

[Ha and Satyanarayanan 2015] Ha, K. and Satyanarayanan, M. (2015). OpenStack++ for Cloudlet Deployment. *School of Computer Science Carnegie Mellon University Pittsburgh*.

[Han et al. 2011] Han, H., Sheng, B., Tan, C. C., Li, Q., and Lu, S. (2011). A Timing-Based Scheme for Rogue AP Detection. *Parallel and Distributed Systems, IEEE Transactions on*, 22(11):1912–1925.

[Hong et al. 2013a] Hong, K., Lillethun, D., Ramachandran, U., Ottenwälder, B., and Koldehofe, B. (2013a). Mobile Fog: A Programming Model for Large-scale Applications on The Internet of Things. In *Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing*, pages 15–20. ACM.

[Hong et al. 2013b] Hong, K., Lillethun, D., Ramachandran, U., Ottenwälder, B., and Koldehofe, B. (2013b). Opportunistic spatio-temporal event processing for mobile situation awareness. In *Proceedings of the 7th ACM international conference on Distributed event-based systems*, pages 195–206. ACM.

[Idland et al. 2015] Idland, E., Øverby, H., and Audestad, J. A. (2015). Economic markets for video streaming services: A case study of netflix and popcorn time. *Norsk Informatikkonferanse (NIK)*.

[Jin et al. 2013] Jin, R., Wang, B., Zhang, P., and Luh, P. B. (2013). Decentralised online charging scheduling for large populations of electric vehicles: a cyber-physical system approach. *International Journal of Parallel, Emergent and Distributed Systems*, 28(1):29–45.

[Kashi and Sharifi 2013] Kashi, S. S. and Sharifi, M. (2013). Connectivity weakness impacts on coordination in wireless sensor and actor networks. *Communications Surveys & Tutorials, IEEE*, 15(1):145–166.

[Khalid et al. 2016] Khalid, M., Yousaf, M. M., Iftikhar, Y., and Fatima, N. (2016). Establishing the State of the Art Knowledge Domain of Cloud Computing. In *Advanced Computer and Communication Engineering Technology*, pages 1001–1014. Springer.

[Klas 2016] Klas, G. I. (2016). Edge Cloud to Cloud Integration for IoT, Y.I Readings - News, Opinions, Analysis. Disponível em: http://yucianga.info/?p=1008. . Acesso em: 20 fev. 2016.

[Lewis et al. 2014] Lewis, G., Echeverría, S., Simanta, S., Bradshaw, B., and Root, J. (2014). Tactical Cloudlets: Moving Cloud Computing to The Edge. In *Military Communications Conference (MILCOM), 2014 IEEE*, pages 1440–1446. IEEE.

[Li and Shimamoto 2012] Li, C. and Shimamoto, S. (2012). An open traffic light control model for reducing vehicles' emissions based on etc vehicles. *Vehicular Technology, IEEE Transactions on*, 61(1):97–110.

[Liu et al. 2015] Liu, K., Ng, J. K. Y., Lee, V. C. S., Son, S. H., and Stojmenovic, I. (2015). Cooperative data scheduling in hybrid vehicular ad hoc networks: Vanet as a software defined network. *IEEE/ACM Transactions on Networking*, PP(99):1–1.

[Luan et al. 2016] Luan, T. H., Gao, L., Li, Z., Xiang, Y., We, G., and Sun, L. (2016). A View of Fog Computing from Networking Perspective. *arXiv preprint arXiv:1602.01509*.

[Madsen et al. 2013] Madsen, H., Albeanu, G., Burtschy, B., and Popentiu-Vladicescu, F. (2013). Reliability in the Utility Computing Era: Towards Reliable Fog Computing. In *Systems, Signals and Image Processing (IWSSIP), 2013 20th International Conference on*, pages 43–46. IEEE.

[Manzalini and Crespi 2016] Manzalini, A. and Crespi, N. (2016). An Edge Operating System Enabling Anything-as -a-Service. *IEEE Communications Magazine*, 54(3):62–67.

[Marforio et al. 2014] Marforio, C., Karapanos, N., Soriente, C., Kostiainen, K., and Capkun, S. (2014). Smartphones as Practical and Secure Location Verification Tokens for Payments. In *NDSS*.

[Mell and Grance 2010] Mell, P. and Grance, T. (2010). The NIST definition of cloud computing. *Communications of the ACM*, 53(6):50.

[Mineraud et al. 2015] Mineraud, J., Mazhelis, O., Su, X., and Tarkoma, S. (2015). A Gap Analysis of Internet-of-Things Platforms. *arXiv preprint arXiv:1502.01181*.

[Mitton et al. 2012] Mitton, N., Papavassiliou, S., Puliafito, A., and Trivedi, K. S. (2012). Combining Cloud and Sensors in a Smart City Environment. *EURASIP journal on Wireless Communications and Networking*, 2012(1):1–10.

[Modi et al. 2013] Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., and Rajarajan, M. (2013). A survey of intrusion detection techniques in cloud. *Journal of Network and Computer Applications*, 36(1):42 – 57.

[Moura and Hutchison 2016] Moura, J. and Hutchison, D. (2016). Review and Analysis of Networking Challenges in Cloud Computing. *Journal of Network and Computer Applications*, 60:113–129.

[Nandyala and Kim 2016] Nandyala, C. S. and Kim, H.-K. (2016). From cloud to fog and iot-based real-time u-healthcare monitoring for smart homes and hospitals. *Atlantic*, 10(2).

[Netflix 2016] Netflix (2016). Netflix open connect. Disponível em: https://openconnect.itp.netflix.com. Acesso em: 15 mar. 2016.

[NetworkWorld 2015] NetworkWorld (2015). Microsoft, interview about Micro Datacentres (MDC). Disponível em: http://www.networkworld.com/article/2979570/cloud-computing/microsoft-researcher-why-micro-datacenters-really-matter-to-mobiles-future.html. Acesso em: 10 fev. 2016.

[Nielsen Norman Group 2014] Nielsen Norman Group (2014). Nielsen's Law of Internet Bandwidth. Disponível em: http://www.nngroup.com/articles/law-of-bandwidth/. Acesso em: 20 dez. 2015.

[Nitti et al. 2015] Nitti, M., Pilloni, V., Colistra, G., and Atzori, L. (2015). The Virtual Object as a Major Element of the Internet of Things: a Survey. *IEEE Communications Surveys and Tutorials*, 99:1–12.

[OpenFog 2016] OpenFog (2016). Open Fog Architecture Overview. Disponível em: http://www.openfogconsortium.org/wp-content/uploads/OpenFog-Architecture-Overview-WP-2-2016.pdf. Acesso em: 16 jan. 2016.

[Openstack 2015] Openstack (2015). Openstack Open Source Cloud Computing Software. Disponível em: https://www.openstack.org/. Acesso em: 20 mar. 2016.

[OpFlex 2014] OpFlex (2014). OpFlex: An Open Policy Protocol White Paper. Disponível em: http://www.cisco.com/c/en/us/solutions/collateral/data-center-virtualization/application-centric-infrastructure/white-paper-c11-731302.html. Acesso em: 20 mar. 2016.

[Orsini et al. 2015] Orsini, G., Bade, D., and Lamersdorf, W. (2015). Computing at the Mobile Edge: Designing Elastic Android Applications for Computation Offloading.

[Ottenwälder et al. 2014] Ottenwälder, B., Koldehofe, B., Rothermel, K., Hong, K., and Ramachandran, U. (2014). Recep: Selection-based reuse for distributed complex event processing. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, pages 59–70. ACM.

[Peng 2004] Peng, G. (2004). Cdn: Content distribution network. *arXiv preprint cs/0411069*.

[Rahimi et al. 2003] Rahimi, M., Shah, H., Sukhatme, G. S., Heideman, J., and Estrin, D. (2003). Studying the feasibility of energy harvesting in a mobile sensor network. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 19–24. IEEE.

[Rao et al. 2012] Rao, B., Saluia, P., Sharma, N., Mittal, A., and Sharma, S. (2012). Cloud Computing for Internet of Things & Sensing Based Applications. In *Sensing Technology (ICST), 2012 Sixth International Conference on*, pages 374–380. IEEE.

[Sarkar et al. 2015] Sarkar, S., Chatterjee, S., and Misra, S. (2015). Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Transactions on Cloud Computing*, PP(99):1–1.

[Satyanarayanan et al. 2009] Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N. (2009). The Case for VM-Based Cloudlets in Mobile Computing. *Pervasive Computing, IEEE*, 8(4):14–23.

[Satyanarayanan et al. 2015] Satyanarayanan, M., Schuster, R., Ebling, M., Fettweis, G., Flinck, H., Joshi, K., and Sabnani, K. (2015). An Open Ecosystem for Mobile-Cloud Convergence. *Communications Magazine, IEEE*, 53(3):63–70.

[Sehgal et al. 2012] Sehgal, A., Perelman, V., Kuryla, S., and Schönwälder, J. (2012). Management of Resource Constrained Devices in the Internet of Things. *Communications Magazine, IEEE*, 50(12):144–149.

[Shah and Pâris 2007] Shah, P. and Pâris, J.-F. (2007). Peer-to-Peer Multimedia Streaming Using BitTorrent. In *Performance, Computing, and Communications Conference, 2007. IPCCC 2007. IEEE Internationa*, pages 340–347. IEEE.

[Shi et al. 2015] Shi, Y., Ding, G., Wang, H., Roman, H. E., and Lu, S. (2015). The fog computing service for healthcare. In *Future Information and Communication Technologies for Ubiquitous HealthCare (Ubi-HealthTech), 2015 2nd International Symposium on*, pages 1–5. IEEE.

[Shimojo et al. 2015] Shimojo, T., Takano, Y., Khan, A., Kaptchouang, S., Tamura, M., and Iwashina, S. (2015). Future mobile core network for efficient service operation. In *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*, pages 1–6.

[Simsek et al. 2016] Simsek, M., Aijaz, A., Dohler, M., Sachs, J., and Fettweis, G. (2016). 5G-Enabled Tactile Internet. *IEEE Journal on Selected Areas in Communications*, 34(3):460–473.

[Stojmenovic 2014a] Stojmenovic, I. (2014a). Fog computing: a cloud to the ground support for smart things and machine-to-machine networks. In *Telecommunication Networks and Applications Conference (ATNAC), 2014 Australasian*, pages 117–122. IEEE.

[Stojmenovic 2014b] Stojmenovic, I. (2014b). Machine-to-machine communications with in-network data aggregation, processing, and actuation for large-scale cyber-physical systems. *Internet of Things Journal, IEEE*, 1(2):122–128.

[Stojmenovic and Wen 2014] Stojmenovic, I. and Wen, S. (2014). The Fog Computing Paradigm: Scenarios and Security Issues. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*, pages 1–8. IEEE.

[Stojmenovic et al. 2015] Stojmenovic, I., Wen, S., Huang, X., and Luan, H. (2015). An overview of fog computing and its security issues. *Concurrency and Computation: Practice and Experience*.

[Stolfo et al. 2012] Stolfo, S. J., Salem, M. B., and Keromytis, A. D. (2012). Fog computing: Mitigating insider data theft attacks in the cloud. In *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*, pages 125–128. IEEE.

[Sudha and Viswanatham 2013] Sudha, S. and Viswanatham, V. M. (2013). Addressing security and privacy issues in cloud computing. *Journal of Theoretical and Applied Information Technology*, 48(2):708–719.

[Suryawanshi and Mandlik 2015] Suryawanshi, R. and Mandlik, G. (2015). Focusing on mobile users at edge and internet of things using fog computing. *International Journal of Scientific Engineering and Technology Research*, 04(17):3225–3231.

[Taivalsaari and Mikkonen 2015] Taivalsaari, A. and Mikkonen, T. (2015). Cloud Technologies for the Internet of Things: Defining a Research Agenda Beyond the Expected Topics. In *Software Engineering and Advanced Applications (SEAA), 2015 41st Euromicro Conference on*, pages 484–488. IEEE.

[Tang et al. 2015] Tang, B., Chen, Z., Hefferman, G., Wei, T., He, H., and Yang, Q. (2015). A hierarchical distributed fog computing architecture for big data analysis in smart cities. In *Proceedings of the ASE BigData & SocialInformatics 2015*, page 28. ACM.

[TinyOS 2016] TinyOS (2016). Main development repository for TinyOS (an OS for embedded, wireless devices). Disponível em: https://github.com/tinyos/tinyos-main. Acesso em: 20 mar. 2016.

[Vaquero and Rodero-Merino 2014] Vaquero, L. M. and Rodero-Merino, L. (2014). Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing. *ACM SIGCOMM Computer Communication Review*, 44(5):27–32.

[Vermesan and Friess 2014] Vermesan, O. and Friess, P. (2014). *Internet of Things - From Research and Innovation to Market Deployment*. River Publishers.

[Wei et al. 2014] Wei, C., Fadlullah, Z. M., Kato, N., and Stojmenovic, I. (2014). On optimally reducing power loss in micro-grids with power storage devices. *Selected Areas in Communications, IEEE Journal on*, 32(7):1361–1370.

[Yao et al. 2013] Yao, D., Yu, C., Jin, H., and Zhou, J. (2013). Energy Efficient Task Scheduling in Mobile Cloud Computing. In *Network and Parallel Computing*, pages 344–355. Springer.

[Yi et al. 2015a] Yi, S., Hao, Z., Qin, Z., and Li, Q. (2015a). Fog Computing: Platform and Applications. In *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*, pages 73–78. IEEE.

[Yi et al. 2015b] Yi, S., Li, C., and Li, Q. (2015b). A Survey of Fog Computing: Concepts, Applications and Issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, pages 37–42. ACM.

[Yi et al. 2015c] Yi, S., Qin, Z., and Li, Q. (2015c). Security and Privacy Issues of Fog Computing: A Survey. In *Wireless Algorithms, Systems, and Applications*, pages 685–695. Springer.

[Yu et al. 2010] Yu, S., Wang, C., Ren, K., and Lou, W. (2010). Achieving secure, scalable, and fine-grained data access control in cloud computing. In *Infocom, 2010 proceedings IEEE*, pages 1–9. Ieee.

[Zao et al. 2014] Zao, J. K., Gan, T.-T., You, C.-K., Chung, C.-E., Wang, Y.-T., Méndez, S. J. R., Mullen, T., Yu, C., Kothe, C., Hsiao, C.-T., et al. (2014). Pervasive Brain Monitoring and Data Sharing Based on Multi-tier Distributed Computing and Linked Data Technology. *Frontiers in human neuroscience*, 8.

[Zaslavsky et al. 2013] Zaslavsky, A., Perera, C., and Georgakopoulos, D. (2013). Sensing as a Service and Big Data. *arXiv preprint arXiv:1301.0159*.

[Zhang et al. 2015] Zhang, B., Mor, N., Kolb, J., Chan, D. S., Lutz, K., Allman, E., Wawrzynek, J., Lee, E., and Kubiatowicz, J. (2015). The Cloud is Not Enough: Saving IoT from the Cloud. In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15)*.

[Zhang et al. 2010] Zhang, Q., Cheng, L., and Boutaba, R. (2010). Cloud Computing: State-of-the-Art and Research Challenges. *Journal of internet services and applications*, 1(1):7–18.

[Zhang et al. 2013] Zhang, W., Tan, G.-Z., and Ding, N. (2013). Traffic Information Detection Based on Scattered Sensor Data: Model and Algorithms. *Adhoc & Sensor Wireless Networks*, 18.

[Zhou et al. 2011] Zhou, B., Cao, J., and Wu, H. (2011). Adaptive traffic light control of multiple intersections in wsn-based its. In *Vehicular technology conference (VTC Spring), 2011 IEEE 73rd*, pages 1–5. IEEE.

[Zikopoulos et al. 2011] Zikopoulos, P., Eaton, C., et al. (2011). *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. McGraw-Hill Osborne Media.

REALIZAÇÃO

computação UFBA

UFBA 70 ANOS

INSTITUTO FEDERAL BAHIA

PROMOÇÃO

LARC

SBC Sociedade Brasileira de Computação

FOMENTO

CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico

CAPES

APOIO

STI 40 anos Superintendência de Tecnologia da Informação | UFBA

POP-BA

PRONEX Filosofia e Ciências

FUNDAÇÃO Pierre Verger

RNP REDE NACIONAL DE ENSINO E PESQUISA

PATROCÍNIO DIAMANTE

nic.br Núcleo de Informação e Coordenação do Ponto BR

cgi.br Comitê Gestor da Internet no Brasil

PATROCÍNIO OURO

acm sigcomm

L3Networks

SOPHOS + arcserve

LogMeIn + KASPERSKY

GRUPO BINÁRIO Integração e Serviços

aruba a Hewlett Packard Enterprise company

PATROCÍNIO BRONZE

Google

IMTECH

Solutis tecnologia

PROTELE engenharia