SBRC 2019

Gramado | RS

XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos

6 a 10 de maio

Gramado | RS

ANAIS DO WGRS 2019

sbrc2019.sbc.org.br

SBRC 2019

Gramado | RS

XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos

6 a 10 de maio

Gramado | RS

ANAIS DO WGRS 2019

sbrc2019.sbc.org.br

EDITORA
Sociedade Brasileira de Computação (SBC)

ORGANIZAÇÃO
Alberto Egon Schaeffer Filho (UFRGS)
Weverton Luis da Costa Cordeiro (UFRGS)
Stênio Fernandes (UFPE)
Luiz Fernando Bittencourt (UNICAMP)
Rafael Lopes Gomes (UECE)
Eduardo Coelho Cerqueira (UFPA)

REALIZAÇÃO
Sociedade Brasileira de Computação (SBC)
Instituto de Informática - Universidade Federal do Rio Grande do Sul (UFRGS)
Laboratório Nacional de Redes de Computadores (LARC)

Copyright ©2019 da Sociedade Brasileira de Computação
Todos os direitos reservados

Capa: Dórian Fogliatto
Produção Editorial: Carlos Raniery P. dos Santos (UFSM)

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

XXIV Workshop de Segurança Cibernética em Dispositivos Conectados (24: 2019: Gramado, RS).

Anais / II Workshop de Gerência e Operação de Redes e Serviços- WGRS; organizado por Alberto Egon Schaeffer Filho, Weverton Luis da Costa Cordeiro, Stênio Fernandes, Luiz Fernando Bittencourt, Rafael Lopes Gomes - Porto Alegre: SBC, 2019

170 p. il. 21 cm.
Vários autores
Inclui bibliografias

1. Redes de Computadores. 2. Sistemas Distribuídos. I. Schaeffer-Filho, Alberto Egon II. Cordeiro, Weverton Luis da Costa III. Fernandes, Stênio IV. Bittencourt, Luiz Fernando V. Gomes, Rafael Lopes VI Título.

Sociedade Brasileira de 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
Renata Mendes de Araujo (UPM), Diretora 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
Ricardo de Oliveira Anido (UNICAMP), Diretor de Relações Profissionais
Esther Colombini (UNICAMP), Diretora de Competições Científicas
Raimundo José de Araújo Macêdo (UFBA), Diretor de Cooperação com Sociedades Científicas
Cláudia Cappeli (UNIRIO), Diretora 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.sbc.org.br

Laboratório Nacional de Redes de Computadores (LARC)

Diretor do Conselho Técnico-Científico
Paulo André da Silva Gonçalves (UFPE)

Vice-Diretora do Conselho Técnico-Científico
Rossana Maria de Castro Andrade (UFC)

Diretor Executivo
Ronaldo Alves Ferreira (UFMS)

Vice-Diretor Executivo
Danielo Gonçalves Gomes (UFC)

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, PUCMG, UNIRIO, UFS e UFU.

Contato
Universidade Federal de Mato Grosso do Sul (UFMS)
Faculdade de Computação
Caixa Postal 549
CEP 79.070-900 Campo Grande - MS - Brasil
http://www.larc.org.br

Organização do SBRC 2019

Coordenadores Gerais
Alberto Egon Schaeffer Filho (UFRGS)
Weverton Luis da Costa Cordeiro (UFRGS)

Coordenadores do Comitê de Programa
Antônio Jorge Gomes Abelém (UFPA)
Fabíola Gonçalves Pereira Greve (UFBA)

Coordenador de Palestras e Tutoriais
Ítalo Cunha (UFMG)

Coordenador de Painéis e Debates
Artur Ziviani (LNCC)

Coordenador de Minicursos
Miguel Elias Mitre Campista (UFRJ)

Coordenador de Workshops
Stênio Fernandes (UFPE)

Coordenador do Salão de Ferramentas
Leandro Villas (UNICAMP)

Coordenador do Concurso de Teses e Dissertações
Daniel Fernandes Macedo (UFMG)

Coordenadores do Hackaton
Raquel Lopes (UFMG)
Luis Carlos de Bona (UFPR)

Comitê de Organização Local
Avelino Zorzo (PUCRS)
Carlos Raniery Paula dos Santos (UFSM)
Cristiano Bonato Both (Unisinos)
Guilherme Rodrigues (IFSUL Charqueadas)
Jéferson Campos Nobre (UFRGS)
Juliano Wickboldt (UFRGS)
Marcelo Caggiani Luizelli (Unipampa)
Marcelo da Silva Conterato (PUCRS)
Rafael Pereira Esteves (IFRS)
Rodrigo Mansilha (Unipampa)
Tiago Ferreto (PUCRS)
Vinícius Guimarães (IFSUL Charquadas)

Comite Consultivo

Fabíola Gonçalves Pereira Greve (UFBA)
Paulo André da Silva Gonçalves (UFPE)
Fábio Luciano Verdi (UFSCAR)
Jó Ueyama (USP)
Antônio Jorge Gomes Abelém (UFPA)
Eduardo Cerqueira (UFPA)
Luiz Fernando Bittencourt (Unicamp)
Rossana Andrade (UFC)
Michele Nogueira (UFPR)
Edmundo Madeira (UNICAMP)

Mensagem dos Coordenadores Gerais

É com grande alegria e orgulho que, após 9 anos, estamos trazendo o Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) novamente para a cidade de Gramado. O SBRC 2019 acontece em um ano que é especial por várias razões: primeiro, o aniversário de 50 anos da ARPANET, a primeira rede a implementar o conjunto de protocolos TCP/IP, que acabaram se estabelecendo mais tarde como os fundamentos da Internet; os 30 anos da invenção da World Wide Web pelo cientista Inglês Tim Berners-Lee; a comemoração dos 30 anos do .br (country code top-level domain), o qual tem atualmente mais de 4 milhões de domínios registrados; por fim, nesse ano comemora-se os 30 anos da Rede Nacional de Ensino e Pesquisa (RNP), que iniciou em 1989 como um projeto de pesquisa. É, portanto, um imenso privilégio termos a oportunidade de sediar o SBRC em um ano que carrega tanto significado para a área de redes de computadores e sistemas distribuídos.

A 37a edição do evento se apoia no histórico de sucesso do SBRC, que tradicionalmente inclui sessões técnicas, minicursos, painéis e debates, workshops, salão de ferramentas e palestras. Além de contar novamente com um Hackathon e com um Concurso de Teses e Dissertações, teremos muitas novidades no SBRC 2019, incluindo as reuniões de mentoria, o evento MUSAS e o workshop de estudantes latino-americanos. As reuniões de mentoria permitem oportunizar conversas 1-a-1 entre estudantes de pós-graduação e pesquisadores de excelência do Brasil e do exterior. O MUSAS (MUlheres em redeS de computadores e sistemAs diStribuídos) visa fomentar conexões entre mulheres atuando nas áreas de redes de computadores e de sistemas distribuídos, no Brasil e no exterior. Já o workshop de estudantes latino-americanos (LANCOMM) serve como um ponto de encontro para estudantes da região apresentarem e discutirem o andamento de suas pesquisas, e obterem feedback construtivo de pesquisadores estrangeiros de grande prestígio.

Neste ano, a trilha principal do SBRC recebeu 228 submissões completas e que entraram no processo de avaliação. Todos os artigos receberam pelo menos 3 revisões, e após um rigoroso processo seletivo, 80 artigos foram aceitos para publicação e organizados em 23 sessões técnicas. O Salão de Ferramentas selecionou 12 trabalhos que a serem demonstrados ao longo do SBRC 2019. Por sua vez, o Concurso de Teses e Dissertações selecionou 8 dissertações de mestrado e 8 teses de doutorado para apresentação durante o evento. Além disso, esse ano o evento conta com 4 palestrantes internacionais, 3 painéis, 1 tutorial, 5 minicursos e 11 workshops.

A organização de um evento com o porte do SBRC é um processo longo, que demanda muita energia e empenho. O evento desse ano só foi possível graças ao apoio e suporte incondicional de muitos grupos de pessoas e instituições. Agradecemos em particular o apoio da SBC, do LARC, do Comitê Consultivo do SBRC e da Comissão Especial de Redes de Computadores e Sistemas Distribuídos da SBC. O evento contou com o apoio do Comitê Gestor da Internet no Brasil (CGI.br) e do Núcleo de Informação e Coordenação do Ponto BR (NIC.br), da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), da Fundação de Amparo à Pesquisa do Estado do Rio

Grande do Sul (FAPERGS) e da ACM SIGCOMM. Contamos novamente também com o apoio institucional da Rede Nacional de Ensino e Pesquisa (RNP). Além disso, o SBRC 2019 recebeu o apoio de importantes patrocinadores, incluindo SAP, HUAWEI, DATACOM, Google, Bedu.tech, BRDigital e AdylNet. Também agradecemos à Universidade Federal do Rio Grande do Sul (UFRGS), e em especial ao Instituto de Informática (INF), pelo suporte incondicional.

Agradecemos a todas as pessoas que trabalharam diretamente na organização do evento. Obrigado pelo excelente trabalho de todos os membros do comitê de organização: Antônio Jorge Gomes Abelém/UFPA e Fabíola Gonçalves Pereira Greve/UFBA (Coordenadores do Comitê de Programa), Stênio Fernandes/UFPE (Coordenador de Workshops), Ítalo Cunha/UFMG (Coordenador de Palestras, Tutoriais e Mentoria), Artur Ziviani/LNCC (Coordenador de Painéis), Miguel Elias Mitre Campista/UFRJ (Coordenador de Minicursos), Leandro Villas/UNICAMP (Coordenador do Salão de Ferramentas), Daniel Fernandes Macedo/UFMG (Coordenador do Concurso de Teses e Dissertações), e Raquel Lopes/UFCG e Luis Carlos De Bona/UFPR (Coordenadores do Hackathon). Agradecemos também aos membros do nosso comitê local, que nos ajudaram na operacionalização de diversas tarefas relacionadas à organização: Avelino Zorzo/PUCRS, Carlos Raniery Paula dos Santos/UFSM, Cristiano Bonato Both/Unisinos, Guilherme Rodrigues/IFSUL Charqueadas, Jéferson Campos Nobre/UFRGS, Juliano Wickboldt/UFRGS, Marcelo Caggiani Luizelli/Unipampa, Marcelo da Silva Conterato/PUCRS, Rafael Esteves/IFRS, Rodrigo Mansilha/Unipampa, Tiago Ferreto/PUCRS, e Vinícius Guimarães/IFSUL Charqueadas. Agradecemos ao Luis Otávio Luz Soares, técnico administrativo do INF/UFRGS, pela imensa ajuda nos mais variados aspectos relacionados à organização do evento, assim como pelo apoio de Leandro Disconzi Vieira e Carlos Alberto da Silveira Junior, também técnicos administrativos da UFRGS. Por fim, agradecemos à nossa equipe local, formada por alunos de graduação, pós-graduação e pós-doutorando da UFRGS, pela dedicação e pronta ajuda sempre que precisamos: Arthur Selle Jacobs, Augusto Zanella Bardini, Bruno Dalmazo, Fernanda da Silva Bonetti, Guilherme Bueno de Oliveira, Guilherme Rotth Zibetti, Isadora Pedrini Possebon, Leonardo Lauryel, Libardo Andrey Quintero González, Lucas Bondan, Lucas Castanheira, Luciano Zembruzki, Mateus Saquetti, Jonatas Marques, Rafael Hengen Ribeiro, e Ricardo Parizotto.

Desejamos a todos um ótimo evento e uma semana produtiva e com muitas trocas de ideias em Gramado.

Alberto Egon Schaeffer Filho e Weverton Luis da Costa Cordeiro
Coordenadores Gerais do SBRC 2019

Mensagem do Coordenador de Workshops

Os Workshops do SBRC são uma importante oportunidade para aprofundamento no conhecimento de temas especializados ou emergentes na nossa comunidade científica. Como em anos anteriores, a coordenação manteve a *Chamada de Propostas de Workshops do SBRC*, que tem estimulado a comunidade brasileira de Redes e Sistemas Distribuídos a discutir a viabilidade de workshops com temas de pesquisa mais tradicionais, bem como estimular o debate para adoção de eventos que tratam de temas mais emergentes.

Tivemos uma diversidade salutar no Comitê de Avaliação em termos de localização geográfica e experiência, gerando, portanto, opiniões complementares sobre as propostas submetidas. Consideramos todas as propostas de altíssima qualidade, com temas, focos e escopo diversos. Nesta edição de 2019, buscamos equilibrar os benefícios trazidos aos participantes com a capacidade de alocação de espaços no evento. Desta forma, foram selecionados 10 Workshops de alta qualidade, além do tradicional Workshop da RNP (*WRNP*). Dentre as propostas aceitas, oito são reedições de workshops tradicionais do SBRC, a saber: **Gerência e Operação de Redes e Serviços** (*WGRS*), **Testes e Tolerância a Falhas** (*WTF*), **Pesquisa Experimental da Internet do Futuro** (*WPEIF*), **Segurança Cibernética em Dispositivos Conectados** (*WSCDC*), **Trabalhos de Iniciação Científica e Graduação** (*WTICG*), **Blockchain: Teoria, Tecnologias e Aplicações** (*WBlockchain*), **Clouds e Aplicações** (*WCGA*), e **Computação Urbana** (*CoUrb*). Como novidade, teremos dois novos workshops, a saber: o Workshop de **Teoria, Tecnologias e Aplicações de Slicing para Infraestruturas Softwarizadas** (*WSlice*) e o **Latin American Student Workshop on Data Communication Networks** (*LANCOMM Student Workshop*), que tem como público-alvo principal os alunos de pós-graduação e pesquisadores da América Latina, além de estreitar os relacionamentos da nossa comunidade com a Association for Computing Machinery (*ACM*).

Como coordenador dos Workshops do SBRC 2019, gostaria de agradecer a todos os envolvidos na seleção das propostas. Primeiramente aos coordenadores gerais do SBRC 2019, Weverton Cordeiro (UFRGS) e Alberto Egon Schaeffer Filho (UFRGS), pelo convite para a coordenação desta chamada de trabalhos, além de todo o apoio necessário para sua condução apropriada. Agradeço também a todos os membros do comitê de avaliação pelo imenso esforço nas revisões e discussões de alta qualidade de todas as propostas submetidas. Por fim, agradeço aos coordenadores dos workshops aceitos, pela dedicação no cumprimento dos prazos e na condução dos trabalhos internos de seleção dos artigos para seus respectivos eventos, mantendo a alta qualidade geral do SBRC. Os inscritos nos diversos workshops esperam ansiosamente pelas apresentações dos trabalhos e pelas frutíferas discussões que serão geradas.

Stênio Fernandes
Coordenador dos Workshops do SBRC 2019

Comitê de Programa

- Alberto Schaeffer-Filho (Federal University of Rio Grande do Sul (UFRGS))
- Aldri dos Santos (Universidade Federal do Paraná)
- Anelise Munaretto (UTFPR)
- Antonio Tadeu Gomes (LNCC)
- Artur Ziviani (LNCC)
- Augusto Neto (Universidade Federal do Rio Grande do Norte - UFRN)
- Bruno Kimura (UNIFESP)
- Bruno Schulze (LNCC)
- Cristiano Both (Universidade do Vale do Rio dos Sinos (UNISINOS))
- Cristiano Silva (Universidade Federal de São João del Rei)
- Daniel Fernandes Macedo (Universidade Federal de Minas Gerais)
- Daniel Guidoni (Universidade Federal de São João del-Rei)
- Denis Rosário (Federal University of Pará)
- Edmundo Madeira (UNICAMP)
- Eduardo Cerqueira (Universidade Federal do Pará - UFPA)
- Fatima Duarte-Figueiredo (PUC Minas)
- Fernando Matos (Federal University of Paraíba)
- Geraldo Pereira (CiC - UnB)
- Islene Garcia (UNICAMP)
- Jeferson Campos Nobre (UFRGS)
- Joaquim Celestino Júnior (State university of Ceará - UECE)
- Joel Rodrigues (National Institute of Telecommunications (Inatel))
- José Augusto Suruagy Monteiro (UFPE)
- Jose-Marcos Nogueira (Universidade Federal de Minas Gerais)
- Juliano Wickboldt (Federal University of Rio Grande do Sul - UFRGS)
- Leandro Villas (UNICAMP)
- Lisandro Zambenedetti Granville (UFRGS)
- Luciano Paschoal Gaspary (UFRGS)
- Luiz Henrique Correia (UFLA)
- Marcelo Marotta (Universidade de Brasília)
- Marcelo Rubinstein (Universidade do Estado do Rio de Janeiro)
- Mauro Fonseca (UTFPR)
- Michele Nogueira (Universidade Federal do Paraná)
- Michelle Wangham (Universidade do Vale do Itajaí)
- Paulo André da Silva Gonçalves (UFPE)
- Rafael Esteves (IFRS)
- Rodolfo Meneguette (Instituto Federal de Educação, Ciência e Tecnologia de São Paulo)
- Roger Immich (Unicamp)
- Ronaldo Ferreira (Universidade Federal de Mato Grosso do Sul)
- Rossana Andrade (Universidade Federal do Ceará - UFC)
- Thiago Genez (University of Cambridge)
- Thiago Henrique Silva (Universidade Tecnológica Federal do Paraná)
- Vinicius Cunha M Borges (Federal University of Goias)

Sumário

Sessão técnica 01 1

Avaliação do Problema de Reordenamento de Pacotes e das Políticas de Escalonamento em Protocolos Multicaminhos 2
Paulo Lenz (Universidade Federal do Paraná – Brazil), Matheus Matos da Silva (Universidade Federal do Paraná – Brazil), Michele Nogueira (Universidade Federal do Paraná – Brazil), Aldri dos Santos (Universidade Federal do Paraná – Brazil)

Policy-Driven Network Traffic Rerouting Through Intent-Based Control Loops 16
Nathan Saraiva (Universidade Estadual de Campinas (UNICAMP) – Brazil), Nazrul Islam (Universidade Estadual de Campinas (UNICAMP) – Brazil), Danny Alex Lachos Perez (Universidade Estadual de Campinas (UNICAMP) – Brazil), Christian Esteve Rothenberg (Universidade Estadual de Campinas (UNICAMP) – Brazil)

Uma Comparação entre os Sistemas de Detecção de Ameaças Distribuídas de Rede Baseado no Processamento de Dados em Fluxo e em Lotes 30
Fabio Schuartz (Universidade Tecnológica Federal do Paraná – Brazil), Anelise Munaretto (Universidade Tecnológica Federal do Paraná – Brazil), Mauro Fonseca (Universidade Tecnológica Federal do Paraná – Brazil)

Seleção Online de Features em Streaming Baseada em Alpha-Investing Para Dados de Ataques DDoS 44
Alissar Moussa (Universidade Federal do Paraná – Brazil), Michele Nogueira (Universidade Federal do Paraná – Brazil), André Guedes (Universidade Federal do Paraná – Brazil)

Sessão técnica 02 58

Uma Abordagem Não Supervisionada para Inferir Qualidade de Experiência em Redes Sem Fio de Grande Escala 59
Diogo Mattos (Universidade Ferderal Fluminense – Brazil), Dianne Medeiros (Universidade Ferderal Fluminense – Brazil), Natalia Fernandes (Universidade Ferderal Fluminense – Brazil), Luiz Magalhaes (Universidade Ferderal Fluminense – Brazil)

Balanceamento de Tráfego entre Servidores de Vídeo MPEG-DASH em Redes Definidas por Software 73
Edenilson Jônatas dos Passos (Universidade do Estado de Santa Catarina – Brazil), Adriano Fiorese (Universidade do Estado de Santa Catarina – Brazil)

Uma Solução para Potencializar a Adaptação e Gerenciamento de Funcionalidades de Infraestruturas Baseadas em SDN 87
Emídio Neto (Instituto Federal do Rio Grande do Norte – Brazil), Felipe Sampaio Dantas Silva (Instituto Federal do Rio Grande do Norte – Brazil), Marcilio Lemos (Universidade Federal do Rio Grande do Norte – Brazil), Augusto Neto (Universidade Federal do Rio Grande do Norte – Brazil)

Uma Solução Flexível e Personalizável para a Composição de Cadeias de Função de Serviço 101
Vinicius Garcia (Universidade Federal do Paraná – Brazil), Marcelo Caggiani Luizelli (Universidade Federal do Pampa – Brazil), Elias Duarte Jr. (Universidade Federal do Paraná – Brazil), Carlos Raniery Paula dos Santos (Universidade Federal de Santa Maria– Brazil)

Sessão técnica 03 115

Um Mecanismo Adaptativo de Descoberta de Serviços para Internet das Coisas Centrado no Consumo Energético 116
Robson Gomes de Melo (Universidade do Estado de Mato Grosso – Brazil), Aldri dos Santos (Universidade Federal do Paraná – Brazil)

Monitoramento e Disseminação Cooperativa de Eventos Emergenciais Apoiado por Agrupamentos de Veículos 130
Jose Everaldo Andrade (Universidade Federal da Paraíba – Brazil), Kevin Veloso (Universidade Federal da Paraíba – Brazil), Nathália Silva (Universidade Federal da Paraíba – Brazil), Aldri dos Santos (Universidade Federal do Paraná – Brazil), Fernando Matos (Universidade Federal da Paraíba – Brazil)

Gerenciamento da Infraestrutura de Comunicação baseado na Distribuição Gaussiana do Tempo de Contato entre Veículos e a Infraestrutura 144
Lucas Silva (Universidade Federal de Ouro Preto – Brazil), Fernanda Sumika Souza (Universidade Federal de São João del-Rei – Brazil), Daniel Guidoni (Universidade Federal de São João del-Rei – Brazil), João F. M. Sarubbi (Universidade Federal de São João del-Rei – Brazil), Cristiano Pitangui (Universidade Federal de São João del-Rei – Brazil), Cristiano Silva (Universidade Federal de São João del-Rei – Brazil)

Análise de Vazamentos Temporais Side-Channel no Contexto da Internet das Coisas 157
Nelson Prates Junior (Universidade Federal do Paraná – Brazil), Andressa Vergutz (Universidade Federal do Paraná – Brazil), Ricardo Tombesi Macedo (Universidade Federal de Santa Maria – Brazil), Michele Nogueira (Universidade Federal do Paraná – Brazil)

XXIV Workshop de Gerência e Operação de Redes e Serviços
SBRC 2019
Sessão Técnica 1

Avaliação do Problema de Reordenamento de Pacotes e das Políticas de Escalonamento em Protocolos Multicaminhos

Paulo Lenz Jr., Matheus D. M. da Silva, Aldri Santos, Michele Nogueira

[1]Núcleo de Redes sem Fio e Redes Avançadas (NR2)
Universidade Federal do Paraná (UFPR)

{pljunior,mdms14,aldri,michele}@inf.ufpr.br

Abstract. *Quality of Service (QoS) and Quality of Experience (QoE) are fundamental to many Internet applications such as streaming services, online games and virtual reality. However, traditional transport-layer protocols are unsatisfactory in these issues due to limitations imposed on their conception. The MPQUIC (MultiPath QUIC) and MPTCP (Multipath TCP) protocols increase transmission performance due to the use of multiple network interfaces available on the devices, allowing to employ multiple and diverse end-to-end paths simultaneously or as redundancy. However, because of the network heterogeneity, multipath end-to-end communication results in challenges, such as out-of-order packet delivery, that reduced its benefits. Hence, we have evaluated the performance of MPQUIC and MPTCP in heterogeneous scenarios through several simulations. Results demonstrate that MPQUIC deals better with packet retransmissions and random losses, however, it is not immune to the out-of-order delivery problem. This work contributes to a better understanding of this problem in multipath transport-layer protocols, promoting future advances.*

Resumo. *A Qualidade de Serviço (QoS) e Qualidade da Experiência (QoE) são fundamentais para muitos aplicativos da Internet, como serviços de streaming, jogos online e realidade virtual. No entanto, os protocolos tradicionais da camada de transporte são insatisfatórios nessas questões devido às limitações impostas em sua concepção. Os protocolos MPQUIC (MultiPath QUIC) e MPTCP (Multipath TCP) aumentam o desempenho da transmissão devido ao uso de múltiplas interfaces de rede disponíveis nos dispositivos, permitindo empregar múltiplos caminhos e diversidade na transmissão fim-a-fim simultaneamente ou de forma redundante. No entanto, devido à heterogeneidade da rede, a comunicação multicaminhos fim-a-fim resulta em desafios, como a entrega de pacotes fora de ordem, que reduz seus benefícios. Por essa razão, avaliamos o desempenho do MPQUIC e MPTCP em cenários heterogêneos através de diversas simulações. Os resultados mostram que o MPQUIC lida melhor com retransmissões de pacotes e perdas aleatórias, mas não está imune ao problema de entrega fora de ordem. Este trabalho contribui para uma melhor compreensão do problema em protocolos multicaminhos, promovendo avanços futuros.*

1. Introdução

As aplicações que demandam acesso à Internet com alta vazão e baixa latência estão cada vez mais presentes na Internet. Os jogos *online*, a realidade virtual, as aplicações médicas e os serviços de *streaming*, como os oferecidos pela Netflix, Amazon Prime e HBO

GO, são alguns exemplos que exigem maior Qualidade de Serviço (QoS) e Qualidade de Experiência (QoE). Com a popularização de dispositivos com diferentes tecnologias de comunicação sem fio (*Multihomed*), que provêm o acesso simultâneo a múltiplas redes, a comunicação fim-a-fim por diferentes caminhos possui o objetivo de oferecer uma maior vazão e menor latência.

Devido às limitações dos protocolos de transporte mais usados na Internet hoje em dia, o TCP e o UDP, diversas abordagens que visam ao uso de múltiplas interfaces de rede simultaneamente foram propostas na literatura. Tais protocolos de transporte são chamados protocolos multicaminhos e aumentam o desempenho da transmissão fim-a-fim razão do uso dos diversos caminhos disponíveis de forma a agregar recursos e, para isto, dependem do algoritmo de controle de congestionamento (CC) e do escalonador de pacotes [Paasch et al. 2014]. O CC tem por objetivo ser amigável com os outros protocolos de modo a transportar dados de maneira justa sem prejudicar outros fluxos ou monopolizar o canal. O escalonador por sua vez deve analisar o estado dos caminhos disponíveis e alocar os dados de acordo com o tamanho máximo da janela de congestionamento.

Dentre os protocolos multicaminhos já padronizados pelo IETF (*Internet Engineering Task Force*), o protocolo Multicaminhos TCP (MPTCP) tem como base o TCP e possibilita a transmissão de dados através de múltiplos caminhos. O Multicaminhos QUIC (MPQUIC) é uma evolução do protocolo QUIC de caminhos únicos desenvolvido pela *Google*. Este protocolo utiliza o UDP para transmissão de dados e permite trabalhar em conjunto com o ambiente legado. Isto diminui a latência das conexões por se abster do *3-way-handshake*. Além disso ele usa a multiplexação de fluxos, tornando os fluxos independentes. Apesar de apresentarem melhor desempenho quando comparados com suas versões de caminho único, já analisadas em [De Coninck and Bonaventure 2017], ambos possuem muitos desafios relacionados à heterogeneidade dos caminhos [Alheid et al. 2016], ao princípio de justiça (*fairness*) [Becke et al. 2012], ao consumo de energia [Kaup et al. 2015], ao custo monetário de transmissão [Secci et al. 2014], à segurança [Pearce and Zeadally 2015], entre outros. Os caminhos heterogêneos degradam o desempenho da transmissão multicaminhos devido à perda e à chegada de pacotes fora de ordem (e.g., problema do reordenamento). O problema do reordenamento aumenta o atraso e reduz a vazão da transmissão multicaminhos, e afeta, principalmente, as aplicações sensíveis ao atraso [Yedugundla et al. 2016].

Desta forma, este trabalho apresenta uma investigação do problema de reordenamento com as atuais políticas de escalonamento dos pricipais protocolos multicaminhos em cenários heterogêneos. A fim de verificar o comportamento do MPTCP e MPQUIC, simulamos diversos cenários com diferentes características e cargas de trabalho. A avaliação demonstra que o número de retransmissões é bastante similar para ambos os protocolos, uma vez que retransmissões implicam em perdas ou atraso na entrega dos pacotes, e faz os segmentos aguardarem reordenação para serem entregues à aplicação. Avaliamos também o número de perdas e o tempo total da conexão. As simulações demonstram que o protocolo MPQUIC lida melhor com as perdas e retransmissões, além de concluir a transferência de dados de forma mais rápida do que o MPTCP, possivelmente por se abster do estabelecimento de conexões, uma vez que está fundamentado no protocolo UDP.

O restante do trabalho está organizado como segue. A Seção 2 apresenta brevemente os conceitos pertinentes ao protocolo QUIC, MPQUIC e MPTCP. A Seção 3 detalha a avaliação realizada. A Seção 4 apresenta as principais estratégias empregadas junto aos métodos de escalonamento para lidar com o problema de reordenamento de pacotes. Por fim, a Seção 5 apresenta as conclusões e direções futuras.

2. Os Protocolos MPTCP, QUIC e MPQUIC

Esta seção apresenta os conceitos e as definições dos protocolos MPTCP, QUIC e MPQUIC. Estes conceitos são necessários para a compreensão destes protocolos. Inicialmente apresentamos o protocolo MPTCP, pela proximidade com o protocolo TCP. Assim como o protocolo MPTCP avança o protocolo TCP mantendo grande parte de suas características, salientamos que grande parte dos atributos do protocolo MPQUIC são herdadas do protocolo QUIC. Desta forma, como o protocolo QUIC não é tão conhecido quanto o TCP, uma descrição do protocolo QUIC antecede a explicação do protocolo MPQUIC.

Os protocolos da camada de transporte gerenciam a comunicação fim-a-fim entre processos de aplicações junto aos *hosts* finais. Dentre eles, os protocolos TCP e UDP são os mais utilizados na comunicação entre cliente e servidor na Internet. O protocolo TCP fornece entrega confiável, ordenada e com verificação de erros. Apesar da sua confiabilidade, ele pode ser inadequado para algumas aplicações que não necessitam do controle exercido por ele. As aplicações que não exigem um serviço de fluxo de dados confiável podem usar o protocolo UDP, adequado para operações em que a verificação e a correção de erros não são necessárias ou são executadas na aplicação. O protocolo MPTCP é baseado no protocolo TCP e o protocolo MPQUIC é baseado no protocolo UDP, porém o MPQUIC aplica mecanismos para garantir a entrega confiável.

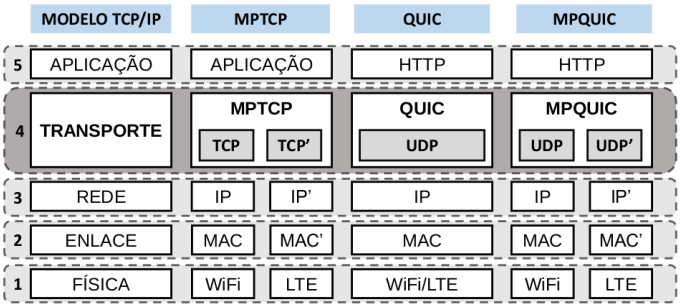


Figura 1. Disposição do protocolo MPTCP, QUIC e MPQUIC no Modelo TCP/IP

Até o momento, os protocolos QUIC e MPQUIC suportam apenas o tráfego web em conjunto com o protocolo da camada de aplicação HTTP, como pode ser observado na Figura 1. Contudo, um de seus objetivos é se tornarem protocolos de transporte generalizados, para múltiplas aplicações. Outro fator importante é o fato desses protocolos (MPTCP, QUIC e MPQUIC) utilizarem o TCP e o UDP sem alterar seus modelos de referências, mantendo-os em consonância com o ambiente legado. De forma simplificada, o MPTCP e o MPQUIC gerenciam o uso das interfaces, podendo iniciar várias conexões com um *host* específico. Para isso, eles administram a utilização de múltiplos fluxos independentes, de forma que um sub-fluxo (*TCP* ou *UDP*) pertença a mesma conexão que outro sub-fluxo (*TCP'* ou *UDP'*). Entretanto, cada um possui controle ao nível de conexão e de fluxo. Consequentemente, as camada inferiores possuem diferentes atributos

para cada canal de comunicação. No nível da camada de rede, cada interface possui um endereço IP diferente (*IP, IP'*). O mesmo ocorre para os endereços de hardware (*MAC, MAC'*), que identificam cada interface utilizada para o tráfego de dados no nível de enlace. Por fim, a comunicação via WiFi e LTE são exemplos das tecnologias utilizadas para propagar os dados, conforme ilustra a Figura 1.

2.1. O protocolo multicaminhos TCP

O protocolo MPTCP possibilita a transmissão de dados através de múltiplos caminhos [Bagnulo 2011], [Silva et al. 2018]. Um dos principais objetivos do MPTCP envolve obter um desempenho superior ao oferecido por um único fluxo TCP, em termos de melhorar a vazão e reduzir a latência. O protocolo MPTCP também possibilita aumentar a resiliência da comunicação, utilizando os caminhos de forma redundante para persistir a conexão em caso de falhas. Uma comunicação MPTCP provê a troca de dados bidirecional entre dois nós comunicando assim como o TCP padrão, não requerendo qualquer mudança na aplicação. Ele também possibilita que os *hosts* finais utilizem diferentes caminhos e endereços para transmitir pacotes pertencentes a uma mesma conexão.

Uma conexão MPTCP inicia de modo similar a uma conexão TCP, utilizando o *3-way handshake*. A Figura 2 ilustra o estabelecimento de uma conexão MPTCP entre cliente e servidor partir dos caminhos C1 e S1, respectivamente. Após a configuração inicial, os *hosts* trocam informações sobre os endereços adicionais e um novo *3-way handshake* é realizado para adicionar um sub-fluxo com os endereços C2-S2. O MPTCP ainda permite que outros sub-fluxos sejam criados, por exemplo, combinando os endereços C1-S2 e C2-B1. Ele considera cada fluxo do TCP como um de seus sub-fluxos, mas que transportam no cabeçalho do TCP um tipo de campo *Option* específico. Todas as operações do MPTCP são sinalizadas através deste campo. Para controlar o envio de pacotes através de diferentes caminhos o MPTCP possui dois níveis de reconhecimento, por sub-fluxo (*SSN - Subflow Sequence Number*) e por conexão (*DSN - Data Sequence Number*), como pode ser observado na Figura 3. Os reconhecimentos SSN, utilizados no TCP, são usados para confirmar o recebimento dos segmentos em cada sub-fluxo independente do DSN.

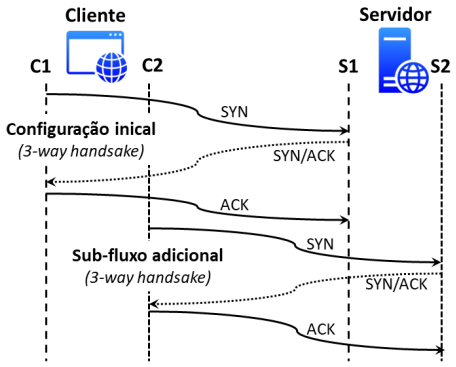


Figura 2. Conexão MPTCP

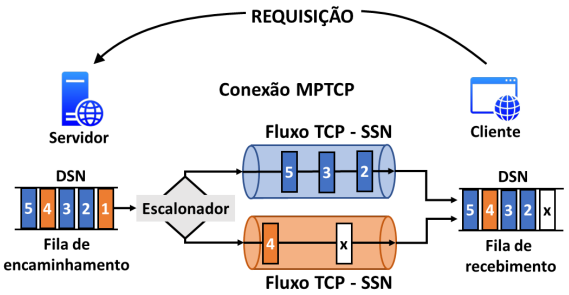


Figura 3. Bloqueio HOL no MPTCP

Apesar dos benefícios que o MPTCP oferece, ele também herda os problemas que o TCP oferece para a conexão fim-a-fim como o bloqueio da fila de recebimento (HOL - *Head-of-line blocking*). O bloqueio HOL ocorre devido ao esgotamento da fila de recebimento em face da entrega de pacotes fora de ordem. Isto causa intermitência no

recebimento dos dados e atrasos na entrega ordenada dos dados à camada de aplicação, como demonstra a Figura 3. Para compensar a heterogeneidade dos caminhos e evitar os problemas com o bloqueio HOL, o protocolo MPTCP usa o mecanismo de retransmissão e penalização [Paasch et al. 2014] , [Possati et al. 2018]. Este mecanismo reencaminha o segmento que possa estar causando o problema do bloqueio HOL no fluxo que tenha espaço na janela de congestionamento. O objetivo é tornar mais rápida a recuperação das situações de bloqueio, compensando a diferença entre os RTT dos caminhos. Contudo, a redução da janela de um fluxo com RTT alto acaba reduzindo e limitando a capacidade de envio e, consequentemente, diminui o desempenho geral da transmissão multicaminhos.

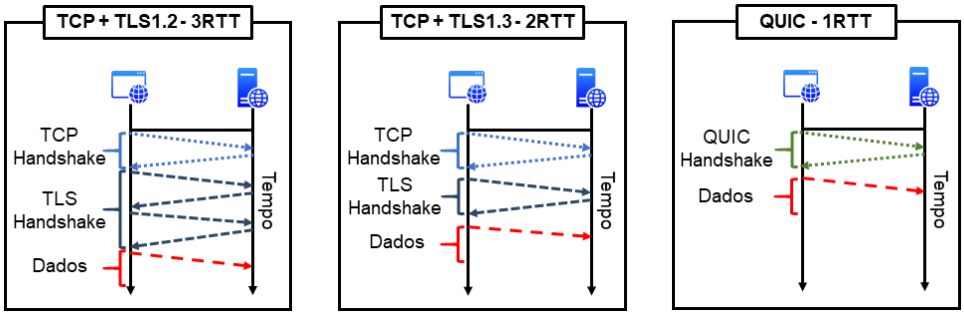
2.2. QUIC

O protocolo QUIC (*Quick UDP Internet Connection*) apresenta um novo conceito de protocolo de transporte. Baseado em técnicas e experiências obtidas com o TCP, SPDY, SCTP, TLS e HTTP/2, o protocolo QUIC utiliza o protocolo UDP em sua essência. Como ele impõe criptografia em todos os seus pacotes, pode ser comparado ao conjunto *TCP + TLS + HTTP/2*. A fim de evitar problemas com o ambiente legado, o QUIC encapsula seus dados em pacotes UDP e aplica mecanismos próprios de controle de conexão e fluxo. Isto facilita a sua implementação, pois não exige alterações no núcleo dos sistemas operacionais ou componentes intermediários (*middleboxes*) à conexão fim-a-fim, e.g., comutadores, *proxies* e *firewalls*. Atualizar ou modificar protocolos consolidados da pilha TCP/IP é uma tarefa difícil e pode levar anos para serem aceitas e distribuídas para os usuários finais. A título de exemplo, o protocolo IP em sua versão 6 levou cerca de 20 anos no processo de desenvolvimento até sua implementação [Dhamdhere et al. 2012].

O pacote QUIC e seu reconhecimento (ACK) contém informações que ajudam o controle do congestionamento e a recuperação de perdas. Cada pacote QUIC é composto de um cabeçalho público (não criptografado) com informações dos *hosts* e da conexão, e dos dados (*payload*) que são sempre criptografados. Esses pacotes transportam um novo número de sequência, incluindo aqueles que transportam dados retransmitidos. Isso elimina a necessidade de um mecanismo separado para distinguir confirmações de retransmissões das transmissões originais, evitando o problema de ambiguidade de retransmissão do TCP. Os reconhecimentos QUIC também codificam explicitamente o atraso entre o recebimento de um pacote e seu reconhecimento sendo enviado e, juntamente com os números de pacote com aumento monotônico, isso permite o cálculo preciso do tempo de ida e volta da rede (RTT). Os quadros ACK do QUIC suportam vários blocos ACK [J. Iyengar and M. Thomson 2018], portanto, o QUIC é mais resiliente à reordenação do que o TCP com suporte ao SACK (sistema de reconhecimento seletivo de pacotes, mensagens ou segmentos), além de poder manter mais bytes em trânsito quando há reordenamento ou perda.

Podemos mensurar os benefícios do QUIC em utilizar o UDP, em vez do TCP, comparando o processo de estabelecimento de uma conexão. Por exemplo, o TCP inicia com a negociação de parâmetros congruentes entre cliente e servidor e esse processo, conhecido como *3-way-handshake*, está presente em todas as conexões desse protocolo. O *handshake* do TCP geralmente leva 1 RTT (tempo de ida e volta) para ser concluído. A fim de aumentar a segurança das transmissões, o protocolo de segurança da camada de transporte (TLS) é empregado para criptografar as mensagens. O TLS também possui um processo de *handshake* para troca de chaves e certificados. Na sua versão mais utilizada

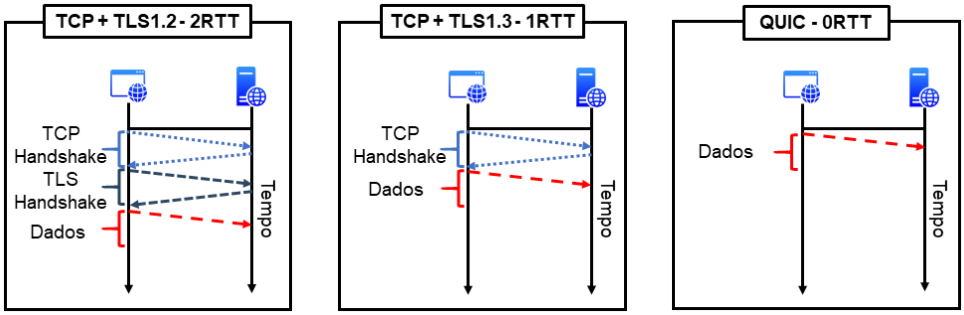
(versão 1.2), ele leva 2 RTTs para efetuar o *handshake* nos casos em que o cliente e servidor não tenham se comunicado anteriormente, totalizando 3 RTTs até o início da transmissão efetiva dos dados, como demonstra a Figura 4(a). A versão 1.3, versão mais atualizada do TLS, diminui seu processo de *handshake* para 1 RTT, além do processo do TCP, totalizando 2 RTTs como ilustra a Figura 4(b). Para o protocolo QUIC, que também criptografa os dados da transmissão e efetua a troca de certificados, o processo de *handshake* leva apenas 1 RTT nos casos em que a comunicação não tenha acontecido previamente, conforme ilustra a Figura 4(c).



(a) TLS v1.2 (b) TLS v1.3 (c) QUIC

Figura 4. Estabelecimento conexão sem comunicação prévia cliente e servidor

Após o estabelecimento da primeira conexão entre cliente/servidor, as chaves e os certificados trocados são armazenados para serem reutilizados. Dessa forma, os protocolos que aumentam a segurança da transmissão diminuem o tempo necessário para estabelecer uma conexão que já foi realizada previamente, desde que essas chaves e certificados não tenham expirado. Assim, como ilustra a Figura 5(a), o TLS na versão 1.2 diminui 1 RTT para estabelecer a conexão a partir da segunda vez que os *hosts* se comunicam. A versão 1.3 necessita apenas do *handshake* do TCP até iniciar a transmissão das requisições do cliente, como mostra a Figura 5(b). O protocolo MPQUIC diferencia-se dos demais por iniciar a transmissão sem a necessidade de um *handshake*, enviando as requisições do cliente no primeiro contato com o servidor, como demonstra a Figura 5(c). O Google estima que 75% das conexões utilizem o 0-RTT *handshake*.



(a) TLS v1.2 (b) TLS v1.3 (c) QUIC

Figura 5. Estabelecimento conexão com comunicação prévia cliente e servidor

2.3. Multicaminhos QUIC

O procolo MPQUIC compartilha todos os benefícios que o QUIC insere na transmissão fim-a-fim com a vantagem de utilizar as múltiplas interfaces de rede disponíveis. Dentre elas, a multiplexação completa de solicitações e respostas, que possibilita a divisão de mensagens em quadros independentes para a transmissão, como pode ser observado na Figura 6. Esta característica reduz os efeitos causados pelo problema de bloqueio HOL bem conhecido para as conexões baseadas no TCP. Por exemplo, se o pacote 1 do caminho 1 for perdido durante a sua transmissão, ele não afetará os outros fluxos compostos pelos pacotes $4, 5$ e $6, 7$. Com a possibilidade de utilizar vários caminhos e a independência dos segmentos que a multiplexação insere, mais informações podem estar em trânsito com menor probabilidade de erros e atrasos causados por perdas de pacotes.

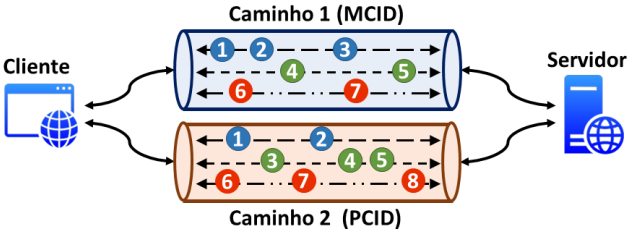


Figura 6. Exemplo de conexão MPQUIC

A administração de múltiplos caminhos exige a adição de novos campos ao cabeçalho público dos pacotes MPQUIC. Durante o estabelecimento de uma conexão, o protocolo MPQUIC executa uma verificação dos caminhos disponíveis entre o cliente/servidor. O caminho pelo qual a conexão é iniciada recebe a identificação MCID (*Master Connection ID*), que é também utilizada para identificar uma conexão. Uma vez que haja apenas um caminho disponível, a conexão seguirá como os padrões do QUIC. Caso haja mais de um endereço utilizável para a transmissão, os *hosts* trocam informações e negociam a adição destes caminhos. Cada caminho adicional recebe então um identificador PCID (*Path Connection ID*) que distingue cada caminho utilizado. O MCID e o PCID compõem a tupla que o protocolo utiliza para gerenciar a conexão em conjunto com o número de sequência do pacote (*PN - Packet Number*), individual para cada caminho.

2.4. Algoritmos de controle de congestionamento e o escalonador

O algoritmo de controle de congestionamento (CC) e o escalonador de segmentos são componentes herdados do MPTCP e adaptados para o MPQUIC. Um dos objetivos do algoritmo de controle de congestionamento é garantir o compartilhamento justo da largura de banda (princípio de *fairness*) e o uso eficiente dos caminhos [De Coninck and Bonaventure 2017]. De forma geral, a extensão desenvolvida para o MPQUIC é mais simples e mais limpa do que a extensão para o MPTCP, e consequentemente, mais fácil de ser implementada. Graças ao suporte para múltiplos fluxos, o MPQUIC não precisa especificar um novo tipo de número de sequência em contraste com o MPTCP. O MPQUIC também não precisa especificar mecanismos para detectar ou reagir à interferência de *middleboxes*, uma vez que todos os dados são criptografados e autenticados. Isso também reduz a possibilidade de ataque a uma conexão MPQUIC comparado com uma conexão MPTCP, cuja segurança depende de chaves trocadas em texto puro durante o aperto de mão inicial (*handshake*).

O algoritmo de controle de congestionamento OLIA (*Opportunistic Linked Increases Algorithm*) [Khalili et al. 2013] é o padrão utilizado tanto para o MPTCP e quanto para o MPQUIC neste trabalho e possui o objetivo de obter melhorias no princípio da eficiência (*Pareto Optimal*). Este princípio estabelece que é impossível aumentar a vazão de uma conexão sem reduzir a vazão de outra ou aumentar o custo de congestionamento. Baseado nesta premissa, o OLIA fornece simultaneamente capacidade de resposta e balanceamento de congestionamento ao adaptar sua janela, que aumenta em função do número de *bytes* transmitidos desde a última perda.

O escalonador é um dos componentes principais de qualquer arquitetura de agregação de banda [Ramaboli et al. 2012]. Para o MPQUIC, o escalonador é um dos responsáveis por alcançar um bom desempenho, dadas as várias restrições dos serviços, requisitos dos usuários e a variação das condições dos caminhos [Paasch et al. 2014]. O escalonador distribui os segmentos através de um ou mais caminhos, com o objetivo que sejam transmitidos, recebidos e entregues de forma confiável e ordenada à aplicação de destino. Os fluxos de dados da aplicação são armazenados na fila de envio e conforme a lógica da política de escalonamento implementada, os pacotes são retirados desta fila e inseridos na fila dos caminhos disponíveis. Atualmente, há duas opções de políticas de escalonamento inseridas no código do MPQUIC: *Round-Robin* (RR) e *Lowest-RTT-First* (Low-RTT). O escalonador RR distribui os dados de modo circular entre os caminhos disponíveis sem critérios de seleção. A política RR em geral não é utilizada por não caracterizar os caminhos e pelo baixo desempenho comparado a outras políticas [Li et al. 2016]. O escalonador Low-RTT seleciona e aloca os dados nos caminhos com menor RTT.

3. Metodologia de Avaliação e Resultados

A construção dos cenários de avaliação está fundamentada nos fatores e parâmetros dos trabalhos de [Paasch et al. 2013] e [De Coninck and Bonaventure 2017]. A Tabela 3 apresenta os valores usados para cada fator. Cada cenário utiliza uma variação dos valores de atraso, fila, largura de banda e taxa de perda para cada caminho de forma independente. Foram construídos 50 cenários distintos com o intuito de simular as diferentes características de redes heterogêneas. Os cenários foram para ambos os protocolos e 3 repetições para cada conjunto. Para a realização dos experimentos, utilizamos um Notebook Dell Inspiron 15 com processador de 1,6 GHz e quatro núcleos, e 8 GB de memória RAM. O Notebook está configurado com o *Ubuntu-16.04 LTS* e com o emulador de redes *Mininet* na versão 2.2.1. O MPTCP foi instalado no *kernel* do *Ubuntu*. A Figura 7 ilustra a representação do cenário dos experimentos composto por dois *hosts* finais (cliente e servidor) e dois comutadores de rede. Com o emulador *Mininet* e a implementação do MPQUIC desenvolvida na linguagem *Go Lang* disponível no *Github* [Clemente 2016].

Fator	Menor Capacidade		Maior Capacidade	
	Min	Max	Min	Max
Capacidade [Mbps]	0,1	100	0,1	100
Atraso (RTT) [ms]	50	100	50	400
Atraso de fila [ms]	0	100	0	2000
Perdas [%]	0	2,5	0	2,5

Tabela 1. Fatores dos caminhos

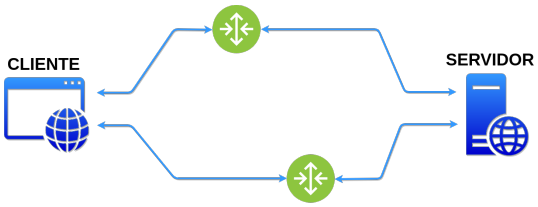


Figura 7. Cenário de teste

Os experimentos simulam a transferência de um arquivo entre um cliente e um servidor. Para avaliarmos o desempenho do escalonador de pacotes *Low-RTT* em cenários e situações mais próximas da realidade, variamos a carga de trabalho em três valores de 300 KB, 1 MB e 20 MB. A primeira carga utilizada simula um tráfego curto, com um arquivo de *300 KB*, referenciado na literatura como *mice flow*. Para a segunda, o arquivo contém *1 MB* e representa conexões mais duradouras comparadas com a primeira. Por fim, a terceira carga de trabalho possui *20 MB* e representa conexões longas, também chamadas de *elephant flow*, com um fluxo contínuo e extenso (em número de *bytes*).

Como métricas para a avaliação, utilizamos a taxa de pacotes retransmitidos ($R = r/s, R \in Emissor$), onde r representa o número total de pacotes retransmitidos e s número total de pacotes enviados observados no emissor. A taxa de pacotes perdidos também é calculada com a simples equação ($L = s - a, s \in Emissor, s \in Recpetor$), onde a representa a quantidade de pacotes recebidos. Por fim, o tempo total da comunicação, isto é, o tempo que compreende o início do processo de *handshake* até o recebimento do ultimo *byte* da transmissão de dados. Note que as retransmissões ocorrem sempre que um pacote é perdido ou após o recebimento de *ACKs* duplicados, isto implica que o segmento aguardará em *buffer* até que o pacote seja retransmitido ou o segmento será descartado após o fim de um temporizador. A Figura 8 ilustra o processo quando há perdas durante a transmissão. Neste exemplo, o pacote 3 é perdido durante a transmissão dos dados. Todos os dados subsequentes que pertença a este segmento são armazenados em *buffer* e aguarda a retransmissão do pacote perdido para que a entrega à aplicação ocorra em ordem.

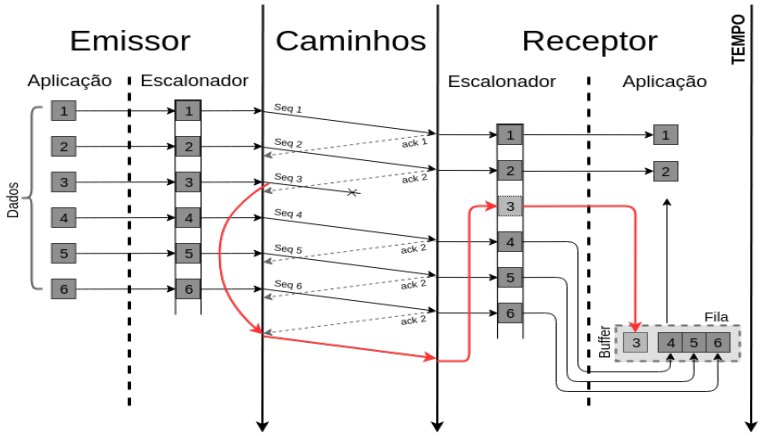
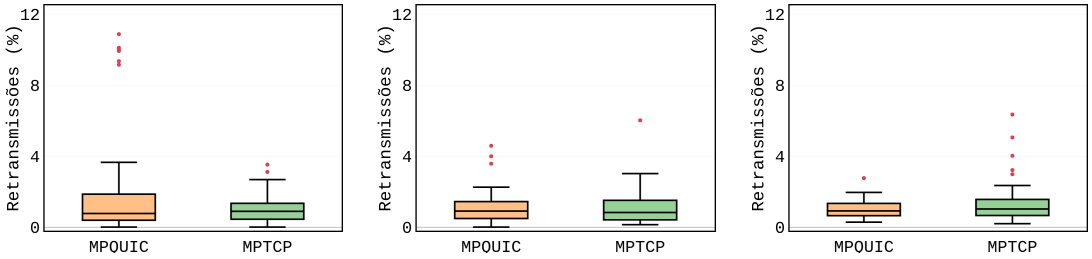


Figura 8. Problema de Reordenamento

A Figura 9(a) ilustra a taxa de retransmissões decorrentes da transferência do arquivo de *300KB* . A distribuição dos dados para o MPTCP apresenta menor variação comparado com o MPQUIC. Também é possível notar uma maior quantidade de pontos discrepantes do MPQUIC comparado com MPTCP, com a máxima em torno de 11%. Contudo, a mediana das amostras avaliadas é menor para o MPQUIC e representa 0,76%, enquanto que para o MPTCP este valor fica em torno de 0,88% das amostras de retransmissão. Nesta ocasião, como a quantidade de dados é relativamente pequena, por vezes o cliente não utiliza o recurso de multicaminhos e realiza toda a transferência por um caminho somente. Esta adversidade foi observada para ambos os protocolos. Alguns trabalhos na literatura estudam este fenômeno e defendem o uso de apenas uma interface de rede

para estes casos, com a justificativa de diminuir o uso de energia, recursos de rede e/ou monetário para os usuários.

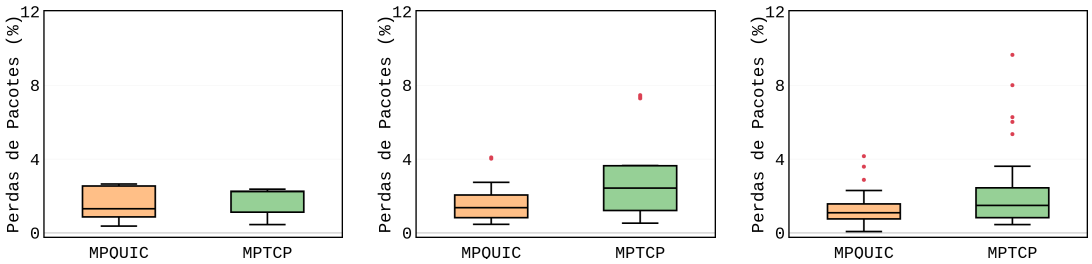


(a) 300 KB (b) 1 MB (c) 20 MB

Figura 9. Retransmissão de pacotes em diferentes cargas de trabalho

A transmissão do arquivo de *1 MB*, como mostra a Figura 9(b), apresenta uma menor distribuição dos dados para o MPQUIC comparado com a carga de trabalho anterior. Contudo, o MPTCP apresenta o mesmo comportamento da carga de *300KB*, com a exceção de um ponto discrepante, o qual representa 6% de pacotes retransmitidos. Para a carga de trabalho de *20MB* , como pode ser visto na Figura 9(c), o MPTCP sofre mais com as retransmissões em relação ao MPQUIC, bem como uma maior incidência de pontos discrepantes. Apesar da baixa taxa de retransmissão observada na avaliação, podemos verificar que o protocolo MPQUIC é moderadamente melhor em lidar com este fenômeno nos cenários em que a carga de trabalho é mais extensa em relação ao MPTCP.

Apesar de estabelecer uma variação de 0 a 2,5% para a taxa de perda aleatória das simulações, este fator pode variar de acordo com outros elementos da rede, como o atraso, tamanho de fila, entre outros. Assim, a Figura 10(a) exibe a variação de perdas sofridas durante a transmissão do arquivo de *300KB* . Ainda que o MPQUIC demonstre uma maior distribuição dos dados com a miníma em 0,37% e a máxima em 2,65%, a mediana se encontra em 1,31% e a média representa 1,51% dos dados observados. Em relação ao MPTCP, a mínima e a máxima representam 0,45% e 2,37%, respectivamente. Contudo, a mediana apresenta 2,26% e a média expõe 1,72% da taxa de pacotes perdidos.



(a) 300 KB (b) 1 MB (c) 20 MB

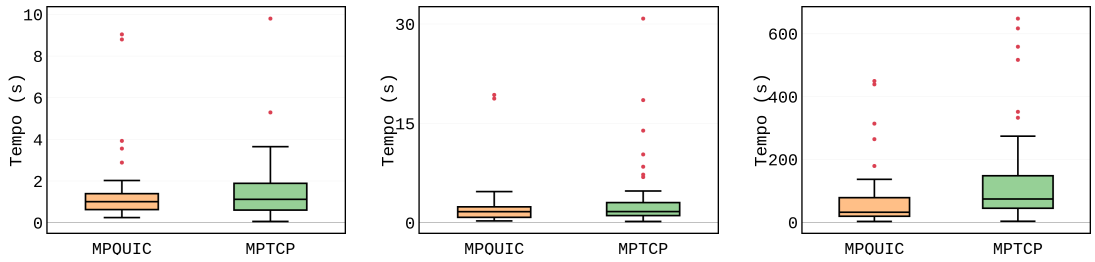
Figura 10. Perdas de pacotes em diferentes cargas de trabalho

Com as cargas de trabalho de *1MB* e *20MB*, o protocolo MPQUIC se sobressai em relação ao MPTCP. De acordo com a Figura 10(b) e 10(c), podemos observar que os resultados obtidos apresentam maior uniformidade e menor ocorrência de pontos discrepantes quando comparado com o MPTCP. No experimento com a carga de *1MB*, o MPQUIC per-

deu 80% menos pacotes em média comparado com o MPTCP. Já com a carga de *20MB*, o MPTCP apresenta 65,6% a mais na média de pacotes perdidos em relação ao MPQUIC.

O MPQUIC leva menos tempo em média na entrega dos dados. Isto se deve a redução da latência durante o estabelecimento da conexão e ao sistema de reconhecimento por pacotes, que oferece informações mais ricas e atualizadas sobre os estados dos *links*. O que colabora com o CC e com o escalonador de pacotes na tomada de decisão. Na Figura 11(a) é possível observar que o MPQUIC mantém a distribuição dos dados observados mais consistentes, entretanto ele apresenta maior quantidade de pontos discrepantes em relação ao MPTCP para a transmissão da carga de *300KB* . Contudo, o MPTCP apresenta o maior valor observado de 9,78 segundos.

Com a carga de trabalho de *1 MB*, o comportamento de ambos os protocolos é bastante similar. Em média, o MPQUIC levou 2,42 segundos e a mediana representa 1,66 segundos. Novamente o MPTCP apresenta a maior variação observada e alcança 30,8 segundos para a transferência do arquivo. Este resultado reflete também a maior capacidade de lidar com as perdas de pacotes analisadas anteriormente, uma vez que com a perda de pacote, a janela de congestionamento é reduzida, diminuindo a quantidade de dados que podem ser transmitidos durante um período de tempo administrado pelo CC.



(a) 300 KB (b) 1 MB (c) 20 MB

Figura 11. Tempo total da conexão

Por fim, o tempo que os protocolos levam para transferir *20MB* evidencia um melhor desempenho na transmissão de dados do MPQUIC, isso pode ser observado na Figura 11(c). Nota-se que a distribuição dos dados referente ao protocolo é menor, comparado com o MPTCP. Isto denota que dentre as simulações realizadas, o MPQUIC apresenta resultados mais consistentes. Em média, o MPQUIC leva 69 segundos para a transferência e sua mediana denota 32 segundos. O MPTCP, em média, precisa de 74,4 segundos para concluir a transferência. Outro ponto consiste nas diferenças entre os valores máximos obtidos, indicando que o MPTCP leva em torno de 200 segundos a mais para concluir toda transferência da carga utilizada, 44,32% mais que o pior caso do MPQUIC.

4. Trabalhos Relacionados

A primeira versão do MPQUIC surgiu em 2017 por [De Coninck and Bonaventure 2017]. Assim, há poucos trabalhos na literatura que abordam o protocolo, e muito menos trabalhos que tratam das políticas de escalonamento para o MPQUIC. Desta forma, nossa referência inicial são os trabalhos sobre escalonamento para o MPTCP, pois em sua atual implementação o MPQUIC herda os mecanismos de controle de congestionamento e escalonamento do MPTCP, tornando-se possível compreender os principais desafios abordados na tarefa de escalonar dados em multicaminhos.

A implementação do escalonador utilizado para o MPTCP e MPQUIC emprega o atraso dos caminhos (RTT) para classificá-los, selecioná-los e alocar os pacotes por eles [Li et al. 2016]. Entretanto, alocar os pacotes pelos caminhos baseados no RTT de cada caminho pode gerar o problema de reordenamento dos pacotes [Garcia-Saavedra et al. 2017]. O atraso de entrega em cada caminho é geralmente incerto e varia ao longo do tempo [Paasch et al. 2014, Albaladejo et al. 2016] devido às filas, retransmissões e perdas de pacotes, políticas do algoritmo de congestionamento e outros [Garcia-Saavedra et al. 2017]. Além das políticas padrões citadas previamente, outras propostas de escalonadores foram desenvolvidas pela comunidade. No trabalho de [Kuhn et al. 2014], os autores apresentam o *Delay-Aware Packet Scheduling* (DAPS) a fim de superar o bloqueio de HOL devido à heterogeneidade dos caminhos. Eles derivaram uma regra geral para o tamanho do *buffer* para o MPTCP. Para isso, o escalonador decide o caminho a enviar cada pacote com base no atraso de envio e na janela de congestionamento individualmente, a fim de prever quando um fluxo enviado por um caminho chegará no destino para enviar outro fluxo esperando que cheguem em sequência.

O trabalho de [Lim et al. 2017] apresenta o escalonador ECF (*Earliest Completion First*), que usa o comprimento do buffer de envio para estimar o tempo total de fluxo (*Flow Complete Time* - FCT) para cada caminho. Se o uso do caminho mais lento aumentar muito o FCT, ele aguardará pelo caminho mais rápido. O ECF toma decisões de escalonamento baseando-se no tamanho da janela de congestionamento e no RTT para evitar períodos de transmissão ociosos e assim alcançar uma maior taxa de transferência agregada. Da mesma forma que o Low-RTT, o ECF prioriza o caminho mais rápido em termos de RTT. Além disso, uma vez que o caminho mais rápido é bloqueado após seu preenchimento pelo controle de congestionamento, ele avalia se é realmente benéfico enviar em um caminho mais lento. Em seguida, ele pode decidir não enviar pelo caminho mais lento, a fim de aguardar a recuperação do caminho mais rápido. O ECF baseia essa decisão na quantidade de dados que aguardam o envio, juntamente com o RTT do caminho e as estimativas de capacidade. Ele não leva em conta o atraso de uma via, portanto, não é capaz de obter uma chegada precisa e ordenada.

[Le and Bui 2018] lida com o problema de reordenamento de pacotes do MPTCP usando um algoritmo de escalonamento baseado no atraso de encaminhamento (FDPS). A ideia principal é que o remetente distribui pacotes através de múltiplos caminhos de acordo com o atraso estimado e a diferença de vazão. Devido à assimetria de latência na Internet, é difícil obter um bom atraso unidirecional. Em geral, as estimativas de atraso consideram as medições do RTT, como feito pelo TCP, assumindo cada atraso, ida e volta, como $RTT/2$. O que nem sempre é aplicável, devido às flutuações do atraso no tempo de ida ser diferente do tempo de volta. Para isso, ele propõe uma metodologia de sincronização de relógio, a fim de se obter o atraso apenas de ida do caminho. Baseado no *timestamp* de pacotes enviados por dois caminhos, o autor calcula as diferenças entre as estatísticas estimadas para se obter o atraso apenas do envio de cada caminho.

5. Conclusão

Este trabalho avaliou o desempenho de transmissões de dados dos protocolos multicaminhos MPTCP e MPQUIC sobre diversos cenários heterogêneos. Diante das diferentes características dos caminhos disponíveis, o MPQUIC apresentou uma pequena vantagem no tempo de transmissão de cargas de trabalho menores. Entretanto, com cargas volu-

mosas, seus benefícios para a transmissão tornam-se mais evidentes, utilizando cerca de 44,32% a menos de tempo que o MPTCP para transmitir *20MB* de dados no pior caso observado. Isto se deve principalmente à redução da latência ao utilizar o protocolo UDP e por se abster do processo de *3-way-handshake* na maioria de suas comunicações. O MPTCP, pelo contrário, exige que todas as interações sejam iniciadas com a negociação de parâmetros da conexão antes de encaminhar as requisições efetivamente ao destinatário. De forma geral, o MPQUIC também lida melhor com as retransmissões e perdas de pacotes durante a comunicação. Graças ao seu sistema de reconhecimento por pacotes e não por blocos como é feito pelo padrão do MPTCP, herdado do mecanismo SACK (*Selective ACKs*) do protocolo TCP. O escalonador de pacotes, em conjunto com o algoritmo de controle de congestionamento, exercem um importante papel na comunicação multicaminhos fim-a-fim. Ainda que o MPQUIC apresente melhores resultados, estes mecanismos são heranças adaptadas do MPTCP, o qual possui diferenças fundamentais quando comparado ao MPQUIC. O desenvolvimento de algoritmos de controle de congestionamento e de escalonadores de pacotes voltados para as características do MPQUIC pode beneficiar ainda mais os usuários e contribuir no alcance de melhores resultados.

Agradecimentos

Os autores agradecem o apoio da UFPR e auxílio financeiro do CNPq e da CAPES, processo # 99999.000404/2016-00.

Referências

[Albaladejo et al. 2016] Albaladejo, M. B., Leith, D. J., and Manzoni, P. (2016). Measurement-based modelling of lte performance in dublin city. In *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, pages 1–6. IEEE.

[Alheid et al. 2016] Alheid, A., Doufexi, A., and Kaleshi, D. (2016). A study on mptcp for tolerating packet reordering and path heterogeneity in wireless networks. In *Wireless Days (WD), 2016*, pages 1–7. IEEE.

[Bagnulo 2011] Bagnulo, M. (2011). Threat Analysis for TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6181, IETF.

[Becke et al. 2012] Becke, M., Dreibholz, T., Adhari, H., and Rathgeb, E. P. (2012). On the fairness of transport protocols in a multi-path environment. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2666–2672. IEEE.

[Clemente 2016] Clemente, L. (2016). A quic implementation in pure go. `https://github.com/lucas-clemente/quic-go`. Acessado em 21/05/2018.

[De Coninck and Bonaventure 2017] De Coninck, Q. and Bonaventure, O. (2017). Multipath quic: Design and evaluation. In *13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT 2017). http://multipath-quic.org*.

[Dhamdhere et al. 2012] Dhamdhere, A., Luckie, M., Huffaker, B., Elmokashfi, A., Aben, E., et al. (2012). Measuring the deployment of ipv6: topology, routing and performance. In *Proceedings of the 2012 Internet Measurement Conference*, pages 537–550. ACM.

[Garcia-Saavedra et al. 2017] Garcia-Saavedra, A., Karzand, M., and Leith, D. J. (2017). Low delay random linear coding and scheduling over multiple interfaces. *IEEE Transactions on Mobile Computing*, 16(11):3100–3114.

[J. Iyengar and M. Thomson 2018] J. Iyengar, E. and M. Thomson, E. (2018). QUIC: A UDP-Based Multiplexed and Secure Transport. Technical report, IETF.

[Kaup et al. 2015] Kaup, F., Wichtlhuber, M., Rado, S., and Hausheer, D. (2015). Can multipath tcp save energy? a measuring and modeling study of mptcp energy consumption. In *Local Computer Networks (LCN), 2015 IEEE 40th Conference on*, pages 442–445. IEEE.

[Khalili et al. 2013] Khalili, R., Gast, N., Popovic, M., et al. (2013). Opportunistic linked-increases congestion control algorithm for mptcp.

[Kuhn et al. 2014] Kuhn, N., Lochin, E., Mifdaoui, A., Sarwar, G., Mehani, O., and Boreli, R. (2014). Daps: Intelligent delay-aware packet scheduling for multipath transport. In *Communications (ICC), 2014 IEEE International Conference on*, pages 1222–1227. IEEE.

[Le and Bui 2018] Le, T.-A. and Bui, L. X. (2018). Forward delay-based packet scheduling algorithm for multipath tcp. *Mobile Networks and Applications*, 23(1):4–12.

[Li et al. 2016] Li, M., Lukyanenko, A., Ou, Z., Yla-Jaaski, A., Tarkoma, S., Coudron, M., and Secci, S. (2016). Multipath Transmission for the Internet: A Survey. *IEEE Communications Surveys & Tutorials*.

[Lim et al. 2017] Lim, Y.-s., Nahum, E. M., Towsley, D., and Gibbens, R. J. (2017). Ecf: An mptcp path scheduler to manage heterogeneous paths. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, pages 147–159. ACM.

[Paasch et al. 2014] Paasch, C., Ferlin, S., Alay, O., and Bonaventure, O. (2014). Experimental evaluation of multipath tcp schedulers. In *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop*, pages 27–32. ACM.

[Paasch et al. 2013] Paasch, C., Khalili, R., and Bonaventure, O. (2013). On the benefits of applying experimental design to improve multipath tcp. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 393–398. ACM.

[Pearce and Zeadally 2015] Pearce, C. and Zeadally, S. (2015). Ancillary impacts of multipath tcp on current and future network security. *IEEE Internet Computing*, 19(5):58–65.

[Possati et al. 2018] Possati, D., Silva, B., Santos, A., and Nogueira, M. (2018). Mitigação de ataque low-rate dos no protocolo mptcp. In *Workshop de Gerência e Operações de Rede (WGRS) - SBRC 2018*.

[Ramaboli et al. 2012] Ramaboli, A. L., Falowo, O. E., and Chan, A. H. (2012). Bandwidth aggregation in heterogeneous wireless networks: A survey of current approaches and issues. *Journal of Network and Computer Applications*, 35(6):1674–1690.

[Secci et al. 2014] Secci, S., Pujolle, G., Nguyen, T. M. T., and Nguyen, S. C. (2014). Performance–cost trade-off strategic evaluation of multipath tcp communications. *IEEE Transactions on Network and Service Management*, 11(2):250–263.

[Silva et al. 2018] Silva, B., Steuck, I., Santos, A., and Nogueira, M. (2018). Escalonador de pacotes para transmissão por multi-caminhos não-correlacionados em redes heterogêneas sem fio. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2018)*.

[Yedugundla et al. 2016] Yedugundla, K., Ferlin, S., Dreibholz, T., Alay, Ö., Kuhn, N., Hurtig, P., and Brunstrom, A. (2016). Is multi-path transport suitable for latency sensitive traffic? *Computer Networks*, 105:1–21.

Policy-Driven Network Traffic Rerouting Through Intent-Based Control Loops

Nathan F. Saraiva de Sousa[1], Nazrul Islam[1], Danny Alex Lachos Perez[1], Christian Esteve Rothenberg[1]

[1]Department of Computer Engineering and Industrial Automation
School of Electrical and Computer Engineering
University of Campinas (UNICAMP)
Campinas, SP, Brazil

{nsaraiva,nazrulis,dlachosper,chesteve}@dca.fee.unicamp.br

Abstract. *Year after year, the growth of video traffic over the Internet keeps increasing. Video streaming over best-effort networks is considered inefficient and inappropriate to meet the expected Quality of Experience (QoE) of the new generation of multimedia services. Over the past few years, a number of technologies have emerged to improve the state of the art of video delivery, including HTTP Adaptive Streaming (HAS) that adapts the bitrate according to network conditions. At the crossroads, Software Defined Networking (SDN) offers options to meet Quality of Service (QoS) objectives for improved video quality by exploiting end-to-end programmability of network behavior. However, traditional SDN approaches require dealing with low-level details from the underlying infrastructure, interfering in the automation and agility of service deployments. To alleviate these issues and overall provide a simpler approach, Intent-Based Networking (IBN) is being proposed to abstract low-level configurations through high-level policy interfaces. In this paper, we explore such an approach by implementing intent-based control loops for video service assurance. The proposed methods dynamically reconfigure the network for service-specific requirements using IBN to define high-level behavior. We experimentally evaluate a use case where video traffic is rerouted based on network conditions to improve the QoS. The Proof-of-Concept results point to the potential of enhancing video content delivery through QoS-aware Intent-based approaches.*

1. Introduction

User demands for Internet video services keep increasing year after year. According to Cisco's Visual Networking Index, 80% of the world's, mobile data traffic will be video by 2019 [Marshall]. As stated in [Cisco 2017], by 2021, 90% of the global consumer traffic and 78% of mobile traffic will be filled with various forms of video. This growth is the main base for the evolution of new multimedia solutions over the networks. An example is HTTP Adaptive Streaming (HAS), widely used around the globe to provide better video quality. While HTTP streaming presents many advantages to deliver improved QoE, managing over the top video streaming applications possess multiple challenges to network designers and service providers [Singh et al. 2012].

Intent-Based networking (IBN) [Clemm et al. 2018] is an emerging concept in the networking community towards practical ways to interface network management and con-

trol systems through high-level policies untangled from underlying infrastructure specificities. A Network Intent refers to the high level of abstraction where the application logic is expressed in terms *what* should be done through high-level semantics and not *how* it should be done. One example of Northbound Interface supporting Intents is the ONOS controller [Linux Foundation 2018], which uses compilation process for translating the Intents to low-level flow rules.

In addition to use high-level abstractions, network operators seek to automate their management and monitoring processes to reduce the Operational Expenditure (OpEx) and to enhance the network performance. The behavior of Control Loops (CL) can be controlled using a policy-based system which automates the process workflows. Through CL, it is possible to improve service efficiency, providing self-healing and service assurance, reducing OpEx, and increasing revenue through shorter time to market [ETSI 2017]. Relevant open source projects such as ONAP [1] and OSM [2] feature CL mechanisms to manage network services lifecycles.

This paper focuses on applying intent-based control loops to improve QoS of video streaming service. Following policy-driven network management (monitor, analyze, and execute) approach, we automate and abstract the details of the underlying network through and effectively a combination of closed control loop and intent-based networking principles. We present a smart control loop mechanism responsible for managing a set of orchestrated actions on the network elements towards assuring the quality of network services. For validation purposes, we developed a proof of concept prototype leveraging best of breed open source tools capable of rerouting video network traffic based on network conditions, high-level policies, and IBN interfaces. Our approach also introduces an abstraction of the network topology through a graph-based database. Targeting an end-to-end video streaming use case, our experimental evaluation presents the obtained network QoS and video QoE metrics (e.g., PSNR and SSIM).

The remainder paper is organized as follows. Section II describes the background and related work in the areas of SDN, closed control loop and IBN. Section III presents the proposed approach for QoS-aware Intent-based Control Loop. Prototype implementation and experimental platform are shown in details in Section IV. Results and Analysis are discussed in Section V. Finally, Section VI concludes the paper with future work directions.

2. Background and Related Work

2.1. Software-Defined Networking and Closed Control Loops

Software-Defined Networking (SDN) allows operators a flexible and efficient utilization of their infrastructures by a software-centric service paradigm [Kreutz et al. 2015]. However, to realize the standard, operators are required to model the end-to-end service. The operator should have the ability to abstract and automates the control of virtual as well as physical resources to deliver the service. The coordinated set of activities behind such process is commonly referred to as orchestration. Network Service Orchestration (NSO) provides multiple network technology [de Sousa et al. 2018] with a common understanding

[1] https://www.onap.org/
[2] https://osm.etsi.org/

and similar alignment. Traditionally, the leading solutions for video service optimizing have focused on the endpoints by adapting the video, source code (and hence throughput) to the varying network conditions [Wu et al. 2001]. SDN is enabled end-to-end network programmability and allows video service delivery with the new approaches based on active participation of the network.

Consequently, network automation through closed control loops is essential for end-to-end service assurance. Four critical phases are necessary to create a simple closed control loop. Indivisibly, each of these steps takes decisions. The four steps are sequenced, *collect=>analysis=>decide=>execute* [Stein 2018]. This approach is assistance in making decisions to reconfigure the network for service-specific requirement dynamically. The network is adjusted with service and resources as well as a new service offering. Policies are used to manage the services and resources to achieve the desired target [ETSI 2017]. For instance, policies can be controlling the new state of the action and overall action is managed by the closed control loop. In a closed control loop, the Machine Learning (ML) and Artificial Intelligence (AI) techniques are used to control and re-configures the network to improve the maintenance and their applications. Big data and machine learning approach require intelligence to handle a large and varying volume of video traffic [Cui et al. 2016]. In turn, inside of the control loop, an adequate policy will trigger a set of orchestrated actions on endpoints to assure the performance of video service concerning QoS. Thus, how to achieve high video streaming performance is a vital issue in the research and development of NSO solutions.

2.2. Intent-based Networking (IBN)

Intent and Intent-Based Networking are emerging trends [Clemm et al. 2018] closely related to Policy, a well-known concept in networking. The key idea around the IBN is to interface the network by means of an "Intent", i.e., no network implementation is specified, the network itself needs to implement all the necessary actions to fulfill the Intent [Noction 2018]. An Intent specifies the "need" rather than the "how". Intent statements can be descriptive or prescriptive [Kiran et al. 2018]. A descriptive (or declarative) Intent uses high-level statements to specify the problem context, for example, "allow traffic between X and Y". Instead, a more network specific information is considered a prescriptive Intent, for example, "from X:10.0.0.1 to Y:10.0.0.2 set rule=allow".

There are different efforts related to building Intent-based systems solutions. ONOS Intent Framework [Linux Foundation b], ODL Network Intent Composition (NIC) [Linux Foundation 2015], ODL Group Based Policy (GBP) [Linux Foundation a], NEMO [c15] are solution examples using a prescriptive approach where users need some knowledge of the network and Intents are controller-specific which means that they depend on the network controller, constraints need to be converted into technology-specific constraints. In case of descriptive frameworks, architectural designs such as DISMI [Sköldström et al. 2017] and iNDIRA [Kiran et al. 2018] consider a simple, high-level, and technology-agnostic interface, which in turn uses a prescriptive module in order to translate advanced primitives into low-level primitives.

DISMI [Sköldström et al. 2017] provides Intent-based northbound interfaces for network controllers that allow applications to request connectivity (including traffic constraints). The DISMI Intent model defines three different classes of primitives: (i) actions (determine the connectivity type), (ii) nouns (determine what to connect), and (iii)

modifiers (determine how the connectivity should behave). Another framework, iNDIRA (Intelligent Network Deployment Intent Renderer Application) [Kiran et al. 2018] interfaces between the SDN north-bound interface and applications, understanding network QoS requirements, and automating the translation for scientific applications. It uses an ontology-based approach to define queries and translates them to network commands.

Other related work on Intent-based frameworks for specific network applications includes [Sanvito et al. 2018, Comer and Rastegarnia 2018, Han et al. 2016, Tsuzaki and Okabe 2017]. The Open Software Defined Framework (OSDF) [Comer and Rastegarnia 2018] provides high-level APIs that allow managers to express network requirements for applications (network configuration, monitoring, and QoS provisioning) without knowing about low-level details (e.g., network topology, flow rule details). In [Han et al. 2016], the authors propose a software-defined Network Virtualization (NV) that automates the network management and configure the virtual network on the basis of Intent. The advantage of this platform is to provide an automated VN management method with high-level Intent. It also allows management applications with high-level representation. According to paper [Sanvito et al. 2018], the authors propose a process to compile the multiple Intents and re-optimize the paths according to their flow statistics. The Intent is handling to optimize the traffic forwarding of ONOS applications. Further, [Tsuzaki and Okabe 2017] describes an automatic management procedure to update the network configuration based on the Intent with reactive configuration.

2.3. Contributions

In short, this work proposes a system that automatically manages the network service in order to assure and optimize video streaming service. Although our focus of this work is video service, the proposed approach could be used for any services that demand restricted quality metrics. Our work envisions different contributions based on 5 key features: *(i)* modularity, use of API to allow add/remove components, *(ii)* adaptive policy, change the policy conditions in runtime, *(iii)* multilevel intent, define levels of intents in a bottom-up approach, *(iv)* topology abstraction, abstraction of the network topology, and *(v)* smart control loop, management using machine intelligence techniques to obtain more knowledge of network service. Table1 compares our proposed to related works.

Table 1. Related works and associated features

Feature	[Sanvito et al. 2018]	[Comer and Rastegarnia 2018]	[Han et al. 2016]	Our proposal
Modularity	YES	NO	YES	YES
Adaptive Policy	YES	YES	NO	YES*
Multilevel Intent	NO	YES	YES	YES
Topology Abstraction	NO	NO	YES	YES
Smart Control Loop	NO	NO	NO	YES*

* *Future work*

3. Proposed Approach: QoS-aware Intent-based Control Loops

Network programmability through paradigms such as SDN and NFV allows for new approaches based on active participation of the network to assure the network service delivery. In support of the decision making to reconfigure the network for service-specific requirements dynamically, a large amount of various input data can be exploited. Due to the dynamicity and large volume of such data, big data and machine-learning approaches

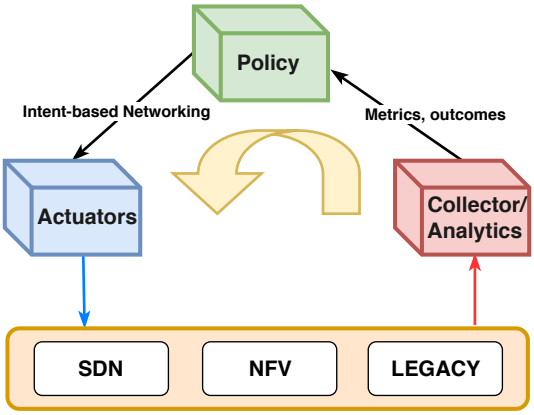


Figure 1. High-level representation of a closed control loop design.

appear as promising approaches for delivering the required intelligence [Cui et al. 2016]. To that end, smart control loop approaches promise to automatically manage a set of orchestrated actions on the network elements (and endpoints if possible) to assure the network service regarding end-to-end quality, fundamentally as perceived by the end-user.

Thus, our objective is to develop a suitable actuation mechanism that implements a closed control loop and translates the user needs and business targets into technology-specific configurations. The system is responsible for assuring the requirements to deliver high-quality video streaming service. The control loop enables the self-healing in end-to-end video services, without human intervention, providing automation and agility in the service management. Figure 1 presents a high-level view of the closed control loop design to monitor and keep the services working based on pre-established agreements, as well as optimize them. In order to abstract the underlying infrastructure through high-level actions and to automate the control loop, we follow principles from IBN.

IBN scopes a higher level of abstraction. For this purpose, we define two layers of intents in a bottom-up approach. In first level, intent assurances network connectivity, e.g., connect two points through different networks. In second level, intents trigger policy-driven actions based on holistic view of service. The Intents enables actions not only on network but also on VNFs and endpoints, including scaling of VNFs, functions placement and changing bitrate.

Briefly, our approach generates a flow of data between the various components that allows implementing an automation system and self-healing of end-to-end network services. In other words, the system detects an issue on a service, and it automatically defines actions to fix it according to pre-set parameters (e.g., SLA). In the next subsections, we explain each component and their functions.

The closed-loop system introduces a delay in monitoring and decision making. However, the system will work with two lines of action: corrective actions, in case of serious failures or SLAs violation, and preventive actions, detected from the Analytics component and avoiding possible future issues.

3.1. Collector/Analytics

The Collector component monitors the deployed service (e.g., video stream) collecting QoS and QoE metrics from SDN and no-SDN data plane. Firstly, it is required to define a set of metrics (e.g., bandwidth, packet loss, CPU) among all those available. Such metrics will enable to register the service performance. After, the data are organized in a well-defined information model for providing service-related information. Differently from works as [da Costa Filho et al. 2018] and [Comer and Rastegarnia 2018], the monitoring of the services is done through the installed intents for each network flow resulting in more precise monitoring.

The collected data are stored in a database to track the historical data and on-line and off-line analysis in runtime. The database stores the status of infrastructure (e.g., topology, components), deployed services and relationship among them. Besides, it must have high performance, scalability, and availability because of the dynamics and the large volume of network data. The same element can also run data analytics based on machine learning (ML) and artificial intelligence (AI) techniques. It analyzes the collected data and exposes parameters and processed information to Policy component. Also, it receives information from performed actions, from there it generates reports and feedback to *Policy* to facilitate the service/infrastructure adjustment. Through historical network data, it is possible to create a tendency curve and an original curve. When a threshold is exceeded in the tendency curve, an alert is sent to the Policy component.

3.2. Policy

Formally, policies are defined as a set of rules that are used to manage and control the state of one or more managed objects. In the control loop context, policies can manage the interaction among services and resources with the environment to obtain the desired objective [ETSI 2017]. There are three types of policies: *(i) Imperative* – uses statements to define the state of targeted objects explicitly. Thus, a set of tasks are executed in the correct order using an event-condition-action tuple; *(ii) Declarative* – expresses the targets of the policy, but not how to achieve it. It follows a formal logic without define clearly the objects state. Examples are SQL and OpenStack Congress[3]; and *(iii) Intent* – expresses the goals of the policy in high-level. It has not formal logic. Thus it needs the translation to one or more primitive policies.

The Policy component is responsible for defining and applying policies that ensure the quality of service. For that, it uses the metrics and information provided by both the Analytics component and database. It abstracts internal configurations as well, i.e., it is unaware of service details, only knows the service Intent and its status. Its operation consists of the monitoring of different metrics and events (e.g., end-to-end data plane video sessions) and reacts by the execution of programmable workflows. The workflow consists of a set of conditions that can be updated in runtime and is triggered by any events (e.g., when the flow has packet loss is greater than 10%). The component always actuates according to networking goals.

We propose a policy that outlines the logic involved in the process of network service management as depicted in Figure 2. It is an extended version from imperative

[3]https://docs.openstack.org/congress/

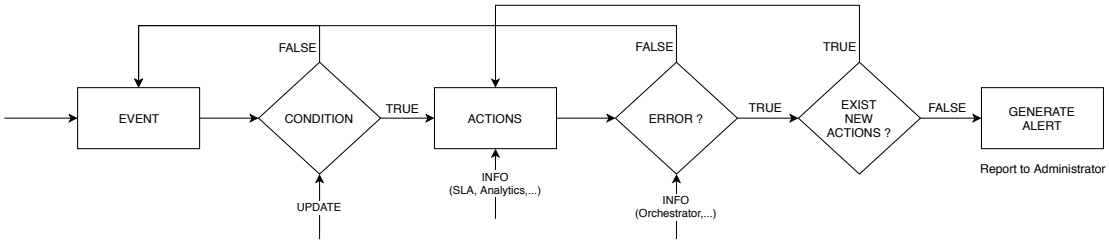


Figure 2. Workflow representation of the proposed extended imperative policy.

policy and describes a simple workflow to implement self-healing in case of error in the execution of the actions. Standard Event-Condition-Action (ECA) policies do not have a detection and recovery mechanism in case of failure to perform actions as we propose.

The workflow starts when an event (e.g., metric threshold) is identified by Policy component. This event is compared to a condition (e.g., traffic is consuming 80% of the link), if it is false, so the system returns to listening state; otherwise, the system lists a set of actions to be performed. The condition can be updated in runtime. After that, the system defines one or more actions to be executed based on information from external elements such as Analytics. If there are no errors in the execution of the operations, the flow returns to the listening state. Otherwise, another action is chosen from the list. This process loops until the system find an action that is executed correctly, or there are no more possible actions. In this case, the system generates an alert reporting to the network administrator.

3.3. Actuators

The Actuator component closes the smart control loop by translating the high-level policies sent by Policy to low-level actions, for instance, OpenFlow flows. The policies are expressed in a specific language closer to the natural based on an Information Model, and require a mapping to network device commands (e.g., API, CLI) of the underlying infrastructure.

Actuators can perform actions in different scopes including SDN, NFV, and Legacy networks. Thus SDN controllers (e.g., OpenDaylight, ONOS), NFV orchestrators or Management Systems can become embodiments of an Actuator. The other components of the architecture are decoupled from the actuators, which are integrated through the northbound interface (NBI). Functional responsibilities of the Actuator include to inform the Policy about the performed actions status, i.e., if the action was executed correctly or not. Given status will enable the Policy to select another action in case of failure and seed the Analytics component towards improving the outcomes.

4. Prototype Implementation and Experimental Platform

Figure 3 illustrates our prototype implementation and experimental platform based on (*i*) Mininet emulated network infrastructure, (*ii*) ElasticSearch to store/access network data, (*iii*) Neo4j graph-based database embodying the annotated topology, (*iv*) Kibana to generate statistical graphs, (*v*) ONOS[4] controller, and (*vi*) proposed IBN/CL Architectures

[4]https://onosproject.org/

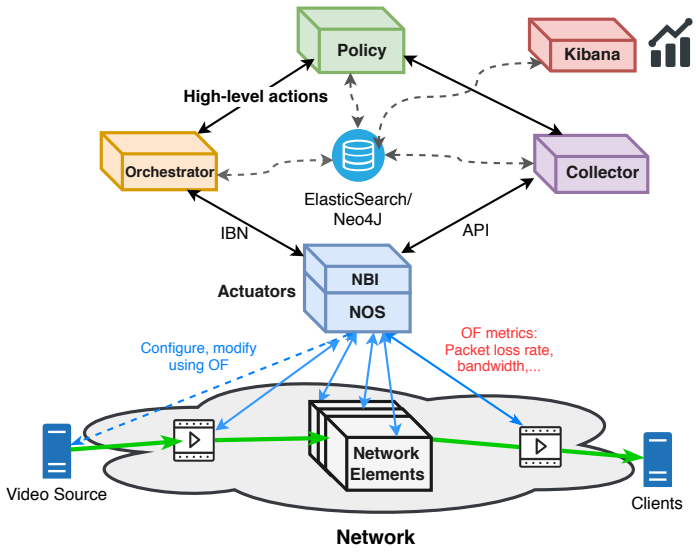


Figure 3. Prototype Implementation and Experimental Platform

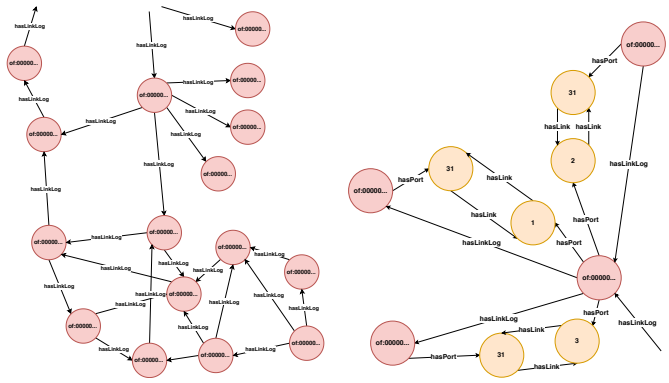
based on the Policy, Collector, and Orchestrator components. Its objective is to reroute network traffic based on network conditions, high-level policies, and IBN interfaces in different scopes (high-and low-level intent). Note that the component Actuator from reference architecture was split into ONOS and Orchestrator. ONOS APIs are used to collect information on the network such as topology, nodes, links, and intent status.

The Collector is a Python-based component responsible for collecting and storing the data from the network. The design requires just one-way measurements of links used by network services. Currently, it collects bandwidth, packet loss, and hop count metrics through Intent Monitor and Reroute (IMR) application from ONOS API. The IMR exposes statistics and re-routing capabilities, of specific Intents, to external applications. There is also a java application that captures topology information and stores them into a graph-oriented database, Neo4J[5]. The topology is stored into Neo4j graph model as shown in Figure 4. In this prototype version, the Analytics component is not implemented.

All network measurements are realized per flow and per link (between switches) and stored into ElasticSearch[6]. Accessible through an extensive and elaborate API, Elasticsearch allows for quick searches in support of data discovery applications. Kibana[7] easies the visualization of Elasticsearch data by means of rich graphics. ElasticSearch and Kibana are highly scalable, well-documented and easy to set up experimental scenario, and it is being used in important projects including ONAP and OSM.

The Policy component is implemented using Python language too. It consumes data directly from ElasticSearch or Collector and realizes a verification of Key Performance Indicators (KPIs). This prototype uses a simple imperative policy, no-extended yet, with three-tuples, consisting of an event, a condition, and an action clause. Pol-

[5]https://neo4j.com/
[6]https://www.elastic.co/elasticsearch
[7]https://www.elastic.co/kibana



(a) Switches (b) Links (Switches + Ports)

Figure 4. Abstracted topology into Neo4J, e.g., switches and links.

icy checks a set of metrics periodically and compares them with some pre-set condition (e.g., packet loss is higher than 5%). If the condition is satisfied, then the Policy sends high-level action to Orchestrator. Examples of high-level action are: *select shortest path avoiding link "X-Y"*, or "*Route traversing links with properties (A, B, C)*."

The Orchestrator, in turn, is a Python-based application that works as a Broker that handles technological details related to network service management. Its function is to translate high-level actions sent by Policy to primitive Intents of ONOS, which are further decomposed into simpler actions (e.g., install OpenFlow flows). It can work as a plugin for the orchestration platform as OSM. Therefore, all actuation is implemented via northbound Intent interface from ONOS. Components consume database information updated at runtime.

5. Experimental Validation and Evaluation

For validation and evaluation through experimentation, we developed a Proof-of-Concept use case scenario based on a video streaming service. We first analyze the connectivity between hosts using `ping`. We then start the video stream from a video server to one or more clients. The Control Loop system monitors the video service and automatically actuates if a problem is found regarding the intended QoS. To this end, the Collector component periodically monitors the bandwidth, packet loss rate, and hop count per flow and per link.

Figure 6 presents the evaluation scenario: network topology composed of 34 devices, organized following an operator-like access-aggregation-core topology, inspired by [da Costa Filho et al. 2018], featuring up to 20 clients connected to the edge switches and one video server. The idea of this scenario is to represent a video server located in the Internet streaming video to client 1 in the access network. VLC [8] is used as the video software.

For the sake of our PoC scenario, the policy is applied over a unique metric: packet loss rate. Considering "Intent" as the desired state of an action to automate the network control, the "Intent" is applied whenever packet loss of monitored flow is greater than X value, e.g., 12%. After identifying the problematic network segment, the system selects

[8] https://www.videolan.org/vlc/index.html

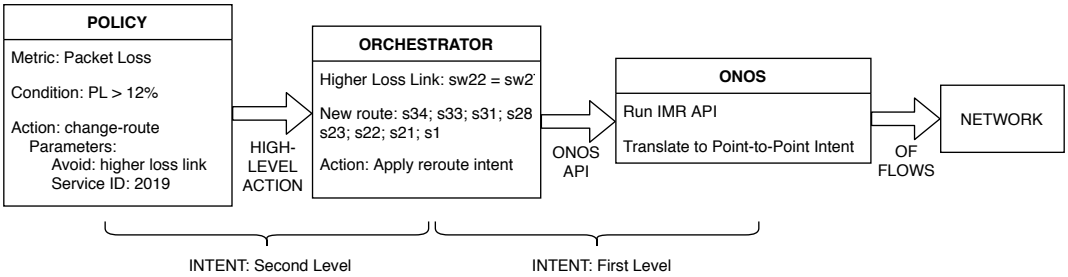


Figure 5. Translation of high-level policies to underlying configurations.

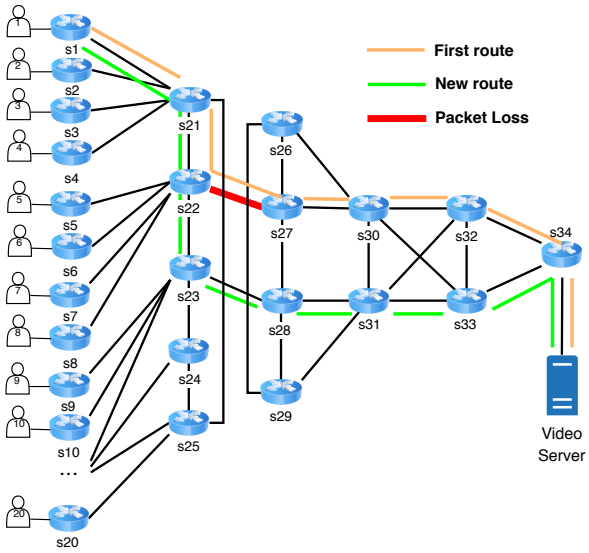


Figure 6. Topology used in the experimental evaluation. Adapted from [da Costa Filho et al. 2018]

the next shortest path avoiding such link. Annotated topology state is kept in the Neo4J database to provide proper routing data. Figure 5 presents detail on the process of policy enforcement until the effective action execution. The Policy sends a high-level action to the Orchestrator to effectively reroute the traffic. The Orchestrator translates the high-level action to Intents supported by the ONOS controller, which, in turn, performs the state changes to adequately reroute the traffic between the video server and the client.

The initial route between the video server and the client (1) goes through on $sw34- > sw32- > sw30- > sw27- > sw22- > sw21- > sw1$. The Control Loop monitors throughput and packet loss metrics. The experiment script uses TC [9] to configure the packet loss to force a 3% loss in the link between $sw22$ and $sw27$. The system identifies the packet loss, but the Policy performs no action since the rate is lower than 12%. Next, the packet loss rate is increased to 13%. In this case, the Policy notes that the threshold was exceeded. After querying ElasticSearch to identify the under-performing link ($sw22 = sw27$), the Orchestrator is requested to reroute the flow to avoid the said link by querying Neo4j to return all shortest routes except that link. The new route is selected and through ONOS APIs the route changed by the IMR and IFWD applications. The new path looks as follows: $sw34- > sw33- > sw31- > sw28- > sw23- > sw22- >$

[9] http://man7.org/linux/man-pages/man8/tc.8.html

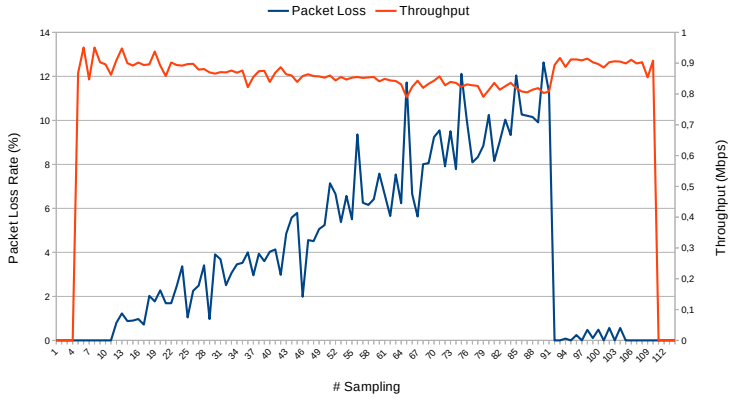


Figure 7. Packet loss and throughput before and after applying the policy.

$sw21- > sw1$. Note that the ONOS Intent framework provides fault-tolerance and resilience even when some switch fails, i.e., the Intent is recompiled by rerouting the flow in the case of failure. If there is no alternative path, the framework tries to establish the path once one becomes available.

5.1. Results and Analysis

Evaluation of QoS metrics In this experiment, the video duration is 3767 seconds and the packet loss in link $sw22 = sw27$ is increased by 1% every 280 seconds. We set up the policy to applying a rerouting action when the flow packet loss is greater than 12% and the polling interval is 35 seconds. Figure 7 presents the observed packet loss and throughput during the execution of the experiment.

Note that the packet loss starts in 0% and gradually increases until it exceeds the limit, around sample #93. Then, the flow is rerouted to another path with no loss, and from that point, the loss drops to practically 0%. Observe that the throughput decreases as losses increases, returning to normal conditions after the policy is applied. This behavior is normal due to the number of packets dropped.

Evaluation of QoE metrics To evaluate the quality of the video, we use the traditional PSNR and SSIM metrics. PSNR can be interpreted as a measure of the peak error whereas SSIM provides a metric of the overall video quality. Video transmissions with higher PSNR and SSIM closer to one (1) indicate better quality of the video. To provide the PSNR and SSIM QoE metrics, we use the Evalvid.[10] The experiment workflow is similar to the previous one. In this case, another video, composed of 15691 frames, is used and the packet loss is manually configured. Figure 8 presents the experiment results, where the X-axis represents the frame number, and the Y-axes represent the PSNR and SSIM.

We can observe no packet loss between frames 1 to 1460 with PSNR ranging between approximately 35 and 50, and SSIM close to 1. The packet loss between 1 to 10 % corresponds to frames 1460 and 7460, where both PSNR and SSIM float and yield low values. After frame number 7460, the policy is applied, and the path is changed to another one without losses, following the noteworthy enhancements in the QoE metrics.

[10]http://www2.tkn.tu-berlin.de/research/evalvid/fw.html

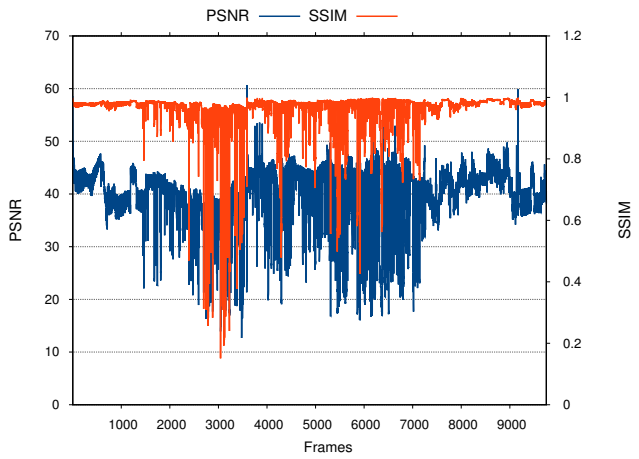
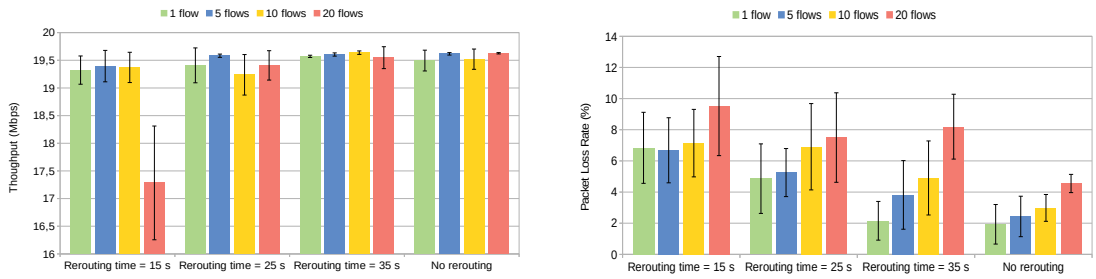


Figure 8. Evaluating QoE metrics: PSNR and SSIM.



(a) (b)

Figure 9. Evaluation of rerouting in terms of (a) Throughput, and (b) Packet loss rate for different flows and rerouting time.

Impact of Control Loop on Network-level Metrics We now analyze the network impact of the closed control loop in case of rerouting events. The goal is to investigate how the network reacts, in terms of packet loss and throughput, when occur rerouting. To this end, we use the same topology of Fig. 6, run *iperf* to generate traffic between *client 1* and the *video Server* and compare traffic in the presence of rerouting to the same traffic without rerouting. A script is used to trigger the route changes between the orange path and the green path. We evaluate the observed behavior for different parallel flows (5, 10 and 20) and different rerouting intervals (15 s, 25 s, and 35 s) over a 350 s simulation. The results are shown in Fig. 9 with 95% of Confidence Interval and ten repetitions.

Regarding throughput (see Fig. 9(a)), the results are statistically equivalent due to the overlapping confidence intervals in all scenarios, except for the case there of 20 simultaneous flows and rerouting interval of 15 seconds. In this case, the frequent changes between the primary and backup paths drop the throughput significantly and with large fluctuation (i.e., high confidence interval). Figure 9(b), presents the packet loss results to assess the impact on performance. The observed behavior is similar to all cases, the larger the amount of parallel flows and lower the rerouting interval, the higher the losses. Various factors influence the performance hit. As one would expect, the crucial one is the

routing changes. The system requires time to apply all new Intents, thereby introducing delay and packet losses in the flows, most critically for large amounts of flows.

6. Conclusions and Future Work

The driving theme of this paper is the potential role of smart control loops and Intent-based Networking concerning automation, abstraction, and self-healing in end-to-end service. We developed a smart control loop approach to improve video service quality. A policy system is exercised to reconfigure the network using IBN as dynamic high-level decisions. The results demonstrated that when the packet loss crosses certain threshold levels, traffic is effectively rerouted thereby enhancing the observed QoS (e.g., Throughput) and QoE metrics (e.g., PSNR, SSIM).

In future work, we will investigate different network topologies and video server locations, including wireless clients using Mininet-WiFi. Parallel work is ongoing on to apply machine learning techniques to identify and classify the traffic to be monitored and improved the smart control loop decisions beyond static packet loss to QoE-centric metrics, including forecast predictions.

Acknowledgment

This research is supported by the Innovation Center, Ericsson S.A., Brazil, grant UNI.64.

References

Nemo - The Application's interface to Intent Based Networks. http://www.nemo-project.net/.

Cisco (2017). Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper - Cisco. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

Clemm, A., Ciavaglia, L., and Granville, L. (2018). Clarifying the Concepts of Intent and Policy. Internet-Draft draft-clemm-nmrg-dist-intent-01, IETF Secretariat. `http://www.ietf.org/internet-drafts/draft-clemm-nmrg-dist-intent-01.txt`.

Comer, D. and Rastegarnia, A. (2018). OSDF: A Framework for Software-Defined Network Programming. *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 1–4.

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.

da Costa Filho, R. I. T., Lautenschläger, W., Kagami, N., Luizelli, M. C., Roesler, V., and Gaspary, L. P. (2018). Scalable QoE-aware Path Selection in SDN-Based Mobile Networks. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 989–997.

de Sousa, N. F. S., Perez, D. A. L., Rosa, R. V., Santos, M. A. S., and Rothenberg, C. E. (2018). Network Service Orchestration: A Survey.

ETSI (2017). Improved operator experience through Experiential Networked Intelligence (ENI). Technical Report 22, ETSI.

Han, Y., Li, J., Hoang, D., Yoo, J., and Hong, J. W. (2016). An Intent-Based Network Virtualization Platform for SDN. In *2016 12th International Conference on Network and Service Management (CNSM)*, pages 353–358.

Kiran, M., Pouyoul, E., Mercian, A., Tierney, B., Guok, C., and Monga, I. (2018). Enabling Intent to Configure Scientific Networks for High Performance Demands. *Future Generation Comp. Syst.*, 79:205–214.

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.

Linux Foundation. Group Based Policy User Guide. https://docs.opendaylight.org/en/stable-fluorine/user-guide/group-based-policy-user-guide.html.

Linux Foundation. ONOS Intent Framework. https://wiki.onosproject.org/display/ONOS/ Intent+Framework.

Linux Foundation (2015). ODL Network Intent Composition. https://wiki.opendaylight.org/view/Network_Intent_Composition:Main.

Linux Foundation (2018). ONOS - A new carrier-grade SDN network operating system designed for high availability, performance, scale-out. https://onosproject.org/.

Marshall, C. By 2019, 80% of the World's Internet Traffic Will Be Video. https://tubularinsights.com/2019-internet-video-traffic/.

Noction (2018). Intent-Based Networking Explained. https://www.noction.com/blog/Intent-based-networking-ibn-explained.

Sanvito, D., Moro, D., Gulli, M., Filippini, I., Capone, A., and Campanella, A. (2018). ONOS Intent Monitor and Reroute Service: Enabling Plug and Play Routing Logic. *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 272–276.

Singh, K. D., Hadjadj-Aoul, Y., and Rubino, G. (2012). Quality of Experience Estimation for Adaptive HTTP/TCP Video Streaming using H.264/AVC. In *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 127–131.

Sköldström, P., Junique, S., Ghafoor, A., Marsico, A., and Siracusa, D. (2017). Dismi - an intent interface for application-centric transport network services. In *2017 19th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4.

Stein, Y. (2018). Closed-loop Implementation – one size does not fit all - TM Forum Inform. https://inform.tmforum.org/digital-transformation-and-maturity/2018/03/closed-loop-implementation-one-size-not-fit/.

Tsuzaki, Y. and Okabe, Y. (2017). Reactive Configuration Updating for Intent-Based Networking. In *2017 International Conference on Information Networking (ICOIN)*, pages 97–102.

Wu, D., Hou, Y. T., Zhu, W., Zhang, Y.-Q., and Peha, J. M. (2001). Streaming video over the internet: approaches and directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):282–300.

Uma Comparação entre os Sistemas de Detecção de Ameaças Distribuídas de Rede Baseado no Processamento de Dados em Fluxo e em Lotes

Fábio C. Schuartz[1], **Mauro Fonseca**[1], **Anelise Munaretto**[1]

[1]CPGEI – Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro, 3165 – 80230-901 – Curitiba – PR – Brazil

phabyo@gmail.com.br, {maurofonseca,anelise}@utfpr.edu.br

Abstract. *With a greater massification of devices connected to the Internet of Things, there is a growing number of vulnerability exploitations detected every year. Thus, faster and more accurate systems are needed to efficiently detect distributed attacks. This paper proposes a system for online detection of distributed network threats using data stream processing. The results obtained by the proposed system are compared with the results obtained by a system using batch processing. The proposed system is evaluated through two metrics: accuracy and number of false-positive and false-negative. The results show that using data stream processing improved detection accuracy by up to 17,50%, reducing the number of false-positives and false-negatives by up to 66,61%.*

Resumo. *Com uma maior massificação de dispositivos conectados à Internet das Coisas, surge um número crescente de ataques de exploração de vulnerabilidades detectados a cada ano. Assim, são necessários sistemas cada vez mais rápidos e precisos para detectar com eficiência ataques distribuídos. Este artigo tem como proposta um sistema de detecção online de ameaças de rede utilizando processamento por fluxos. Os resultados obtidos pelo sistema proposto são comparados com os resultados obtidos por um sistema utilizando processamento por lotes. O sistema proposto é avaliado através de duas métricas: acurácia e número de falsos-positivos e falsos-negativos. Os resultados mostram que utilizar o processamento por fluxo de dados melhorou a acurácia de detecção em até 17,50%, reduzindo o número de falsos-positivos e falsos-negativos em até 66,61%.*

1. Introdução

Com a expansão das redes de computadores mundialmente, as pessoas estão cada vez mais conectadas, utilizando dispositivos que compartilham informações através da Internet. Este crescente aumento de dispositivos em funcionamento acarreta em uma gama de vulnerabilidades que podem ser exploradas por um agente malicioso. A Internet das Coisas (*Internet of Things* - IoT) proporciona a conectividade entre milhões de dispositivos globalmente e isso permite que uma vulnerabilidade possa afetar esses aparelhos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

simultaneamente. Um ataque de varredura de portas pode descobrir uma vulnerabilidade que, futuramente, permite um ataque de negação de serviço (*Denial of Service* - DoS) em larga escala. De acordo com o relatório de ameaças divulgado pela empresa Symantec [Symantec 2018], a Internet das Coisas continua a crescer como alvo primário para os cibercriminosos explorarem. O número de ataques à IoT cresceu de aproximadamente 6.000 em 2016 para 50.000 em 2017, um aumento de 600% em apenas um ano, segundo o relatório. Atualmente, bilhões de dispositivos que utilizam a IoT estão em uso, e a companhia de pesquisa e análises Gartner [Gartner 2016] estima que mais de 50% dos novos processos e sistemas de negócios incluirão um componente ligado à IoT em 2020. Ainda, a empresa F5 Labs [Labs 2017] divulgou seu relatório global sobre ameaças à IoT, o qual examinou como atacantes desenvolveram *botnets* para visar dispositivos conectados especificamente à IoT, os quais cresceram 280% em relação ao relatório semestral anterior.

Com isso em mente, é importante termos mecanismos de segurança que possam detectar essas formas de ataque de uma maneira rápida e com acurácia para termos uma Internet das Coisas segura. Entretanto, as técnicas comuns de detecção para os ataques de varredura e negação de serviço não são eficientes e rápidas o suficiente para trabalhar com um grande volume de dados de tráfego. Assim, tem se tornado mais comum o uso de algoritmos de aprendizagem de máquina para detectar tais intrusões, pois a alta capacidade de processamento distribuído por máquinas utilizando o processamento de fluxo permite construir algoritmos complexos e eficientes que podem tratar os dados de maneira *online*.

Este artigo propõe uma comparação entre os métodos de classificação de dados através de um sistema de aprendizado e classificação em lotes em relação a um sistema de aprendizado e classificação por fluxos, utilizando o sistema de detecção de intrusão para classificar tráfego distribuído de rede, proposto por Schuartz et al. [Schuartz et al. 2017]. A principal contribuição deste trabalho é demonstrar que o uso de processamento de dados em fluxo apresenta melhores resultados comparados ao processamento de dados em lotes, comumente utilizado nos diversos trabalhos na literatura. O sistema proposto é avaliado quanto a acurácia e número de falsos-positivos e falsos-negativos sobre um banco de dados de tráfego de rede simulada, amplamente testada pela comunidade científica. Os resultados obtidos mostram que a utilização do processamento em fluxos traz ganhos consideráveis para a detecção de ataques distribuídos de varredura de portas e de negação de serviço, sem deteriorar a detecção *online*.

O restante deste artigo é dividido da seguinte maneira. Na Seção 2 está a discussão dos trabalhos relacionados. A Seção 3 apresenta o sistema proposto de detecção de ataques. A Seção 4 apresenta o conjunto de dados de tráfego rotulados utilizado na classificação. A Seção 5 apresenta os métodos de classificação e de seleção de características que foram utilizados para a avaliação. A Seção 6 apresenta os resultados obtidos e discute os mesmos. Por último, a Seção 7 conclui o artigo.

2. Trabalhos Relacionados

A literatura apresenta diversas técnicas para detecção de ameaças utilizando aprendizado de máquina. Essas abordagens de aprendizado de máquina podem ser classificadas em quatro tipos de aprendizados: supervisionado, não-supervisionado, semi-

supervisionado e por reforço. O aprendizado supervisionado ocorre quando o conjunto de dados estiver rotulado, ou seja, cada amostra contém a saída (classificação) correta que ela deve gerar, enquanto no aprendizado não-supervisionado as amostras não possuem esse rótulo. O semi-supervisionado contém amostras com e sem rotulação, e no aprendizado por reforço os agentes tentam encontrar a melhor maneira de atingir uma meta, recebendo uma recompensa quando realizam uma ação que os deixem mais perto do objetivo [Pecht and Kang 2019]. Na área de redes, utiliza-se o aprendizado supervisionado para realizar a classificação de ataques, enquanto o não-supervisionado é utilizado para detectar padrões e anomalias. Na classificação de ataques, os métodos mais utilizados na literatura são as árvores de decisão, o K-Vizinho Mais Próximo (*K-Nearest Neighbor* - KNN), as redes Bayesianas e as Máquinas de Vetores de Suporte (*Support Vector Machine* - SVM) [Chen et al. 2015].

Corrêa et al. apresentam um estudo sobre a utilização da técnica de mineração de dados conhecida como Árvore Adaptativa de Hoeffding (*Hoeffding Adaptive Tree* - HAT) para criar um modelo de previsão com objetivo de detectar intrusões em uma rede [Corrêa et al. 2017]. Os resultados mostram uma acurácia no modelo de previsão do HAT em torno de 95%, com tempo de processamento de cada amostra na base de dados sendo aproximadamente 10^{-5}s. Desale et al. discutem técnicas de classificação e melhoria de desempenho em sistemas de detecção de intrusão utilizando diferentes classificadores [Desale et al. 2015]. Os resultados gerados pela análise dos classificadores mostram que Naive Bayes e Árvore de Hoeffding apresentam os melhores resultados. Em termos de acurácia, Naive Bayes apresenta melhor resultado, porém leva mais tempo para classificar que a Árvore de Hoeffding.

No trabalho apresentado por Lobato et al., é proposto um sistema de detecção de ameaças em tempo real por processamento de dados utilizando a arquitetura Lambda [Lobato et al. 2016]. Os resultados mostram uma acurácia com valores acima de 95% e taxa de falsos-positivos abaixo de 6%, analisando até 3,57 milhões de amostras por minutos. Schuartz et al. propõe uma plataforma distribuída para detecção de ameaças em tempo real por análise de fluxos [Schuartz et al. 2017]. O sistema proposto trabalha com diferentes algoritmos de aprendizado de máquina em cima de uma plataforma distribuída, permitindo diversos classificadores operarem simultaneamente. Os resultados obtidos mostram uma acurácia média acima de 90% e tempo de detecção aproximado de $95\mu s$ por fluxo.

Embora os trabalhos apresentados anteriormente apresentem uma técnica de detecção em tempo real, os classificadores são inicialmente preparados através de uma base de dados de treinamento e depois utilizados para classificar um fluxo de dados, de maneira estática. Este artigo propõe um sistema distribuído de detecção de ameaças de rede utilizando algoritmos de processamento em fluxo, onde o modelo aprendido está constantemente sendo atualizado para refletir as novas amostras que chegam pelo fluxo de dados. Os dados que chegam são classificados e incorporados à base de dados, atualizando os modelos de classificação e permitindo os classificadores trabalharem sobre as mudanças de relacionamento entre dados de entrada e saída com o decorrer do tempo (*concept drift*).

3. O Sistema de Classificação Proposto

O sistema proposto para detecção de ameaças de rede em tempo real é dividido em cinco módulos que englobam os sensores para a coleta de dados em diversas fontes, a normalização das informações coletadas, o transporte dos dados normalizados para o sistema de processamento em fluxo, a análise do fluxo de dados em tempo real e a visualização dos resultados obtidos e de alertas que sejam gerados. Os sensores são responsáveis pela coleta dos dados e os demais módulos estão contidos em uma nuvem dedicada para processamento distribuído de dados, conforme proposto em [Schuartz et al. 2017].

O módulo de coleta de dados é composto por ferramentas de coleta de tráfego executadas em sensores distribuídos na rede. Os sensores podem ser máquinas virtuais em ambientes virtualizados ou máquinas físicas com espelhamento de tráfego de um *link* da rede. Esses dados são coletados através da captura dos pacotes Ethernet que entram pelas interfaces do sistema e de arquivos de *log* do sistema operacional e outras aplicações.

O módulo de normalização das informações obtêm características de fluxo através da extração de dados dos cabeçalhos dos pacotes, utilizando janelas de tempo bem definidas. Os fluxos são definidos por uma sequência de pacotes que possuem um mesmo IP de origem e um mesmo IP de destino, com o objetivo de detectar ameaças de rede, como varredura de portas e ataques de negação de serviço. A partir dos cabeçalhos dos pacotes TCP/IP de cada fluxo, obtêm-se 84 características, tais como a quantidade de portas de origem e destino, e número de pacotes TCP, UDP e ICMP.

Para transportar os dados de um ponto ao outro é utilizado um agente de mensagens, isto é, uma ferramenta para a publicação e assinatura (*publish and subscribe*) de uma fila de mensagens. Cada fonte produz dados normalizados e os publica em uma fila. Cada unidade de processamento em fluxos escolhe quais filas assinar, podendo ele escolher uma ou mais filas.

O módulo de processamento em fluxos classifica as amostras recebidas, uma por uma, através de diferentes algoritmos de aprendizado de máquina. Antes da classificação, um filtro de características é aplicado às amostras recebidas com o objetivo de reduzir a complexidade do processamento. O filtro é definido através de algoritmos de redução de dimensionalidade e seleção de características pré-estabelecidos. A definição do modelo de classificação é realizada de forma *online*, através do processamento da amostra recebida e da atualização de sua estrutura de dados contendo a nova amostra, sem exceder o limite de memória e de maneira mais rápida possível.

O módulo de visualização recebe as informações obtidas na camada anterior e fornece uma interface entre o sistema e o usuário. O módulo mostra a análise da rede em tempo real e gera alertas de tráfego suspeito, caso necessário.

3.1. Mineração de Fluxo de Dados

Um fluxo de dados pode ser definido como uma sequência contínua e mutável de dados que constantemente chega em um sistema para ser processado ou armazenado. Os dados são massivos (por exemplo, em volumes de *terabytes*), temporalmente ordenados, variando rapidamente e potencialmente infinito. Essas características causam desafios no campo de fluxo de dados [Kholghi et al. 2010]. Os métodos tradicionais de mineração de

dados normalmente requerem múltiplos exames dos dados e por isso são inviáveis para as aplicações de fluxo de dados.

Devido ao grande volume de dados gerado pelas redes de comunicação e tráfego de Internet, e ao alto custo de armazenamento, é impossível armazenar completamente todos os fluxos de dados ou examiná-los múltiplas vezes. Assim, é necessário utilizar técnicas para extrair as informações embutidas nos dados. A mineração do fluxo de dados (*Data Stream Mining* - DSM) é caracterizada por permitir o processamento de dados em tempo real através do conceito de processamento incremental, onde o novo dado é analisado assim que é recebido.

A DSM refere-se à extração da estrutura de conhecimento que é representada como modelos e padrões dos fluxos infinitos de informações. Devido a limitação dos fluxos de dados, os métodos propostos na literatura são baseados em estatísticas, cálculo e teorias da complexidade. Os dados de entrada (origem da informação) pode ocorrer de diversas fontes. Uma porção dos dados de entrada é selecionada utilizando técnicas tais como amostragem ou agregação. Estes dados são então entregues a um algoritmo que é responsável em atualizar o modelo de predição. Por último, o resultado da classificação é conhecido para cada amostra processada.

O fluxo de entrada $a_1, a_2, ...$ chega sequencialmente, item por item, e descreve um sinal subjacente A, uma função unidimensional

$$A : [1...N] \rightarrow R^2 \tag{1}$$

onde a_i são incrementos de $A[j]$. Considere

$$a_i = (j, I_i), I_i \geq 0 \tag{2}$$

e

$$A_i[j] = A_{i-1}[j] + I_i \tag{3}$$

onde A_i é o estado do sinal após observar o i-ésimo item no fluxo. Múltiplos a_i podem incrementar um dado $A[j]$ com o tempo. O objetivo é que o fluxo de dados A seja processado e para cada amostra a_i existe uma classe predita associada com ela. Isso pode ser representado pela tupla (a_i, b_i), onde a classe predita é representada por b_i. Assim, a predição será correta caso $a_i = b_i$ e incorreta caso $a_i \neq b_i$.

Este modelo de fluxo de dados é o mais popular e utilizado para monitoramento de endereços IP que acessam um servidor *web*, endereços IP de origem que enviam pacotes em um *link*, etc., pois o mesmo endereço pode acessar o servidor *web* múltiplas vezes ou enviar múltiplos pacotes pelo *link* com o tempo [Kholghi et al. 2010].

3.2. Características Utilizadas

No agrupamento de fluxos de rede, os pacotes de rede são abstraídos em fluxos na camada de rede, onde um fluxo é caracterizado como a sequência de pacotes que vai de um endereço IP de origem para o mesmo endereço IP de destino, durante uma janela

fixa de tempo. Para este tipo de agrupamento, as características originais inferidas são variáveis numéricas que estão relacionadas ao conjunto completo de todos os pacotes transmitidos entre dois endereços de IP.

Inicialmente são extraídas um total de 84 características do tráfego de rede a partir de um arquivo *pcap*, através do extrator de características baseado em fluxos *CICFlowMeter* [CICFlowMeter 2017], [Lashkari et al. 2017], incluindo IP de origem, porta de origem, IP de destino, porta de destino e protocolo. Todas as características são definidas e explicadas no site da ferramenta *CICFlowMeter* [CICFlowMeter 2017].

A seguir, são encontrados os conjuntos contendo as melhores características para detectar cada tipo de ataque, dentre as 84 características extraídas inicialmente. Em seu trabalho, Sharafaldin et al. apresenta os conjuntos contendo as melhores características para cada tipo de ataque [Sharafaldin et al. 2018]. Os ataques detectados neste artigo são três: *Botnet*, *PortScan* e *DDoS*. As vantagens da redução de dimensionalidade podem ser encontradas no trabalho de Kezih e Taibi [Kezih and Taibi 2013]. A Tabela 1 mostra a lista das melhores características selecionadas para detecção de cada ataque.

Tabela 1. Melhores características para detecção dos três tipos de ataque (*Botnet*, *PortScan* e *DDos*), segundo [Sharafaldin et al. 2018].

Tipo de Ataque	Características
Botnet	Subflow F. Bytes Total Len F. Packets F. Packet Len Mean B. Packets/s
PortScan	Init Win F. Bytes B.Packets/s PSH Flag Count
DDoS	B. Packet Len Std Avg Packet Size Flow Duration Flow IAT Std

Conforme mostrado na Tabela 1, para detectar um ataque do tipo *Botnet*, as melhores características são encaminhando de *bytes* de sub-fluxo (*Subflow Forwarding Bytes*), encaminhamento do comprimento e média dos pacotes (*Total Length Forwarding Packets* e *Forwarding Packet Length Mean*) e pacotes de retorno (*Backward Packets per Second*). Para ataques do tipo *PortScan*, utilizam-se os *bytes* da janela de encaminhamento inicial (*Initial Windows Forwarding Bytes*), pacotes de retorno (*Backward Packets per Second*) e *push flags* (*PSH Flag Count*). Por último, os ataques de *DDoS* são melhores detectados utilizando as características de comprimento do pacote de retorno (*Backward Packets Length Standard*), tamanho médio dos pacotes (*Average Packet Size*), e dois intervalos de tempo de chegada (*Flow Duration* e *Flow IAT Standard*).

4. Conjunto de Dados Utilizado

Com o objetivo de avaliar o desempenho da abordagem proposta para detecção de anomalias, foram realizados experimentos utilizando um conjunto de dados disponibilizado publicamente na Internet. Dos conjunto de dados existentes, a maioria está

desatualizada e não são mais confiáveis. Alguns dos conjuntos sofrem de uma falta de diversidade e volume no tráfego, outros não cobrem a variedade dos ataques conhecidos enquanto outros possuem conjunto de características e *metadata* incompletos [Sharafaldin et al. 2018].

O conjunto de dados CICIDS2017 [Sharafaldin et al. 2017] contém tráfego legítimo e os ataques atuais mais comuns encontrados na literatura. O conjunto de dados utilizado refere-se ao dia sete de julho de 2017, onde ataques do tipo *Botnet*, *PortScan* e *DDos* ocorreram em horários específicos pré-determinados. O CICIDS2017 consiste de fluxos de rede rotulados, incluindo o *payload* completo dos pacotes, no formato *pcap*, os perfis correspondentes com os fluxos rotulados, e arquivos *CSV* para fins de aprendizado de máquina e aprendizado profundo.

5. Algoritmos de Classificação

Nesta seção são apresentados alguns algoritmos de mineração de dados em fluxos mais utilizados na classificação de tráfego com aprendizado de máquina [Buczak and Guven 2016]. Para classificar as ameaças, três classificadores de características distintas são utilizados: árvore de decisão (*Decision Tree*), árvore adaptativa Hoeffding (*Hoeffding Adaptive Tree*) e bayesiano simples (*Naive Bayes*). A árvore adaptativa Hoeffding é um algoritmo baseado em árvore de decisão, enquanto o bayesiano simples é um algoritmo baseado em probabilidade. Esses classificadores visam estimar a eficiência de classificadores simples e de baixo processamento, e são capazes de realizar o treinamento do modelo de classificação com grande rapidez, permitindo a rápida detecção de ameaças. Os algoritmos foram escolhidos devido sua larga utilização e por seus comportamentos bem definidos na detecção de intrusão [Buczak and Guven 2016].

5.1. Árvore de Decisão - (Decision Tree - DT)

A árvore de decisão é um dos principais métodos de mineração de dados. Uma árvore de decisão padrão consiste de uma raiz, um número de galhos, um número de nós e um número de folhas. Um galho é uma cadeia de nós, da raiz à folha, e cada nó apresenta uma característica do sistema [Sun 2010]. Cada nó armazena os valores de probabilidade de cada classe e são utilizados como parâmetros na tomada de decisões. Quando uma amostra desconhecida entra no modelo, o algoritmo percorre os nós, iniciando pela raiz, avaliando os atributos e calculando a probabilidade daquela amostra pertencer a uma classe específica. Assim, economiza-se tempo na classificação, pois normalmente não é necessário percorrer todas as características da amostra para ocorrer a classificação da mesma. A árvore é gerada inicialmente pela caracterização do nó raiz, contendo a probabilidade de cada classe na amostragem. A seguir, o nó é dividido, gerando filhos que representam novas características da amostragem e que estão associados a um conjunto de probabilidades da classe relativo a essa característica. O processo é reiterado para todos os nós até ser atingido a probabilidade de 100% para alguma classe, sendo este um nó folha. O caminho para cada folha explica o motivo e resultados para cada amostra. Baseado na estrutura da árvore, certas regras podem ser obtidas e a correlação entre amostras é encontrada para futuras classificações.

O algoritmo C4.5 é amplamente usado para classificar um conjunto de dados desconhecidos, onde os valores das características são testados na árvore de decisões. O procedimento do algoritmo C4.5 é detalhado a seguir [Sun 2010].

Passo 1: Calcular a entropia da informação para identificar a classe no conjunto de treinamento S.

$$Info(S) = -\sum_{i=1}^{k}\{[freq(C_i, S/[S])]log_2[freq(C_i, S/[S])]\} \tag{4}$$

onde $[S]$ é o número de amostras no conjunto de treinamento. C_i é uma classe, $i = 1, 2, ..., k$. k é o número de amostras e $freq(C_i, S)$ é o número de amostras inclusas em C_i.

Passo 2: Calcular o valor da informação esperada, $Info_x(S)$, do teste X à partição S.

$$Info_x(S) = -\sum_{i=1}^{k}[(S/[S])Info(S_i)] \tag{5}$$

onde i é o número de saídas para teste X e S_i é o subconjunto de S correspondente a i-ésima saída.

Passo 3: Calcular o ganho de informação após a partição de acordo com teste X.

$$Gain(X) = Info(S) - Info_x(S) \tag{6}$$

Passo 4: Calcular o valor da partição $SplitInfo(X)$ para a partição S em i subconjntos.

$$SplitInfo(X) = \sum_{i=1}^{k}[(S/[S])log_2(S/[S])] \tag{7}$$

Passo 5: Calcular a taxa de ganho de informação de $SplitInfo(X)$ em $Gain(X)$.

$$GainRation(X) = Gain(X)/SplitInfo(X) \tag{8}$$

O $GainRation(X)$ compensa pela quantidade de informações providas pelo conjunto de treinamento. Assim, o atributo com maior $GainRation(X)$ é colocado como raiz da árvore de decisão.

Passo 6: O modelo da árvore de decisão pode ser construído repetindo do **passo 1** ao **passo 5** para cada galho.

5.2. Árvore Adaptativa Hoeffding - (Hoeffding Adaptive Tree - HAT)

O algoritmo Hoeffding é um algoritmo de aprendizado por árvore de decisão, também conhecido como *Very Fast Decision Tree* (VFDT), e que apresenta características importantes para processamento de dados em fluxos. A árvore de decisão é gerada incrementalmente e o conteúdo das amostras coletadas são conhecidas dinamicamente durante o processamento. O VFDT possui estruturas compactas, não necessitando armazenar os

dados processados, apenas informações estatísticas relevantes. Ainda, cada amostra é processada apenas uma vez.

O VFDT utiliza uma métrica estatística chamada limite Hoeffding. Este limite apresenta informações sobre quando o grupo é grande o suficiente para representar uma população que não é inteiramente conhecida ainda [Hoeffding 1963]. O limite Hoeffding possui um fator de erro estimado que pode ser parametrizado, definido por:

$$\varepsilon = \sqrt{\frac{R^2 log(1/\delta)}{2n}} \tag{9}$$

onde R é uma variável aleatória pertencente ao conjunto de valores reais, n é o número de amostras processadas, δ é um valor configurável que descreve a margem aceitável do erro de cálculo.

Entretanto, o algoritmo VFDT não possui a habilidade de detectar e adaptar ao *concept drift* existente no processamento de fluxo de dados. Gama et al. mostra que uma sequência de dados processados ao decorrer do tempo tem seu padrão de comportamento alterado [Gama et al. 2014]. Bifet and Gavaldà [Bifet and Gavaldà 2009] adaptaram o algoritmo VFDT para detectar o *concept drift* e ajustar o modelo de previsão. Esta técnica é conhecida como árvore adaptativa de Hoeffding (*Hoeffding Adaptive Tree* - HAT).

O HAT utiliza uma janela para definir qual dado é relevante aos nós de decisão, definida através de estimações estatísticas coletadas durante o processamento das amostras. Com o tempo, as amostras mais antigas são descartadas pelo algoritmo e as novas amostras atualizam o modelo de conhecimento, adaptando, assim, o modelo para classificar o novo padrão de dados.

5.3. Bayesiano Simples - (Naive Bayes - NB)

O algoritmo bayesiano simples (*Naive Bayes* - NB) é um classificador probabilístico, onde cada característica do sistema não tem influência no valor das demais características, simplificando assim a predição de classificação e tornando-se robusto ao ruído nos dados de entrada. O método calcula inicialmente as probabilidades para cada característica (ou para um conjunto de características) corresponder a uma determinada classe, através dos dados de treinamento. Quando uma nova amostra desconhecida chega ao sistema, o algoritmo calcula a probabilidade de pertencer a cada uma das classes, para cada característica. A probabilidade da amostra pertencer a cada uma das classes é resultado do produto de todas as probabilidades de cada característica. A classe que obtiver a maior probabilidade estimada é escolhida para a classificação da amostra.

6. Avaliação e Resultados

Para avaliar a acurácia da classificação, um protótipo do sistema proposto foi implementado em um sistema composto por uma máquina com processador Intel Core i7-4770S a 3.10 GHz, com 8 GB de memória RAM e 1 TB de disco rígido, utilizando o sistema operacional Ubuntu v16.

O conjunto de características utilizado para classificação dos métodos de aprendizado de máquina foram escolhidos de acordo com os tipos de ataque contidos no conjunto

de dados utilizado. A partir das 84 características iniciais, foram selecionadas 10 características, consideradas as melhores para detectar os ataques, conforme apresentado em [Sharafaldin et al. 2018]. Os métodos de aprendizado de máquina utilizados são: árvore de decisão (DT), árvore adaptiva Hoeffding (HAT) e algoritmo bayesiano simples (NB). Os algoritmos foram implementados em linguagem Java. Para realizar a classificação em lotes, utilizaram-se as bibliotecas da ferramenta Weka [Hall et al. 2009]. Na classificação em fluxos, foram utilizadas as bibliotecas da ferramenta MOA [Bifet et al. 2011].

Na classificação em lotes, o treinamento dos algoritmos foi realizado através do algoritmo *K-fold* de validação cruzada, visando melhorar a generalização dos modelos e evitar, assim, o *overfitting* dos dados. A validação cruzada consiste em separar o conjunto de dados em k partições aleatórias sucessivamente, onde serão utilizadas $k - 1$ partições no treinamento e a partição restante é utilizada na validação do modelo. Após k iterações, o modelo terá utilizado todo o conjunto inicial e é possível avaliar as estatísticas de validação dos modelos através da média de todos as iterações, proporcionando maior confiança no resultado. Foi utilizado para este trabalho, o valor de k igual a 10 iterações. O conjunto de dados inicialmente foi balanceado igualmente entre tráfego normal e ameaças, evitando classificação tendenciosa dos dados em função da classe predominante.

Na classificação em fluxos, os métodos tradicionais de avaliação de mineração de dados, como o *K-fold* de validação cruzada, não pode ser usado para validação dos algoritmos de mineração de fluxo de dados (*Data Stream Mining* - DSM) [Gama et al. 2013]. O DSM apresenta características como dados potencialmente infinitos, modelos de previsão adaptivos e grandes quantidades de informações. Bifet et al. [Bifet et al. 2010] utiliza o conceito de *RAM-hora* para avaliar o desempenho, onde a quantidade de memória utilizada pelo modelo de predição é armazenada durante o processamento. Gama et al. [Gama et al. 2009] propuseram um método de avaliação para DSM chamado método sequencial preditivo (*Predictive Sequential Method - Prequential*). Neste método de avaliação, o número de amostras classificadas incorretamente é acumulada dentro de uma janela w do total de amostras processadas. O erro e classificação pode ser calculado a cada instante i através da equação:

$$P_e(i) = \frac{1}{i} \sum_{k=1}^{i} F(y \neq y') \tag{10}$$

A partir do método sequencial preditivo é possível calcular diversas medidas estatísticas para classificação do fluxo de dados. A ferramenta MOA oferece diversas métricas de validação, tal como a acurácia. Neste trabalho, utilizam-se a acurácia e a matriz de confusão para avaliar os métodos de classificação por processamento em lotes e em fluxos.

As Tabelas 2 e 3 apresentam os resultados de acurácia obtidos pelos métodos de classificação avaliados para o conjunto de dados, considerando os cenários onde todas as características foram utilizadas e utilizando um conjunto reduzido de características.

Os resultados mostram que reduzir o número de características aumentou a acurácia dos classificadores entre 0,48% e 6,82%, sendo o bayesiano simples o maior afetado pela redução das características. Entretanto, observou-se que os algoritmos de

Tabela 2. Acurácia de classificação do conjunto de dados utilizando algoritmos de classificação em lotes.

Classificador	84 Características	10 Características
Árvore de Decisão	97,4601%	97,9299%
Árvore Adaptiva Hoeffding	90,2387%	90,8732%
Bayesiano Simples	70,7794%	75,6124%

Tabela 3. Acurácia de classificação do conjunto de dados utilizando algoritmos de classificação em fluxos.

Classificador	84 Características	10 Características
Árvore de Decisão	98,7254%	98,9683%
Árvore Adaptiva Hoeffding	95,4019%	95,8556%
Bayesiano Simples	85,9081%	88,8470%

aprendizado, em média, executaram as classificações seis vezes mais rápido utilizando apenas 10 características. Os algoritmos de processamento em fluxo apresentaram ganho na acurácia de 1,06%, 5,48% e 17,50% para os algoritmos árvore de decisões, árvore adaptiva Hoeffding e bayesiano simples, respectivamente, em comparação ao processamento em lotes. Observa-se que em ambos os métodos de classificação em lotes e por fluxos, a redução da dimensionalidade de 84 para 10 características aumenta a acurácia nos três métodos de classificação. Porém, mesmo reduzindo a dimensionalidade do método em lotes, o processamento em fluxos apresenta melhores resultados utilizando o espaço total de características. Com a redução para 10 características no método por fluxos, os resultados apresentam ainda maior acurácia.

As Tabelas 4, 5, 6, 7, 8 e 9 apresentam as matrizes de confusão dos métodos de classificação avaliados, considerando o conjunto de 10 características principais.

Tabela 4. Matriz de confusão do algoritmo Árvore de Decisão, utilizando classificação em lotes e 10 características principais.

Árvore de Decisão	Normal	Botnet	PortScan	DDoS
Normal	**1.010.171**	18	193	205
Botnet	48.736	**940.165**	7.701	13.985
PortScan	5.224	2.339	**997.793**	5.231
DDoS	47	1	0	**1.010.539**

Tabela 5. Matriz de confusão do algoritmo Árvore de Decisão, utilizando classificação em fluxos e 10 características principais.

Árvore de Decisão	Normal	Botnet	PortScan	DDoS
Normal	**1.010.390**	7	92	98
Botnet	21.399	**980.665**	3.314	5.209
PortScan	1.988	1.108	**1.005.160**	2.331
DDoS	26	0	0	**1.010.561**

Tabela 6. Matriz de confusão do algoritmo Árvore Adaptiva Hoeffding, utilizando classificação em lotes e 10 características principais.

Árvore Adaptiva Hoeffding	Normal	Botnet	PortScan	DDoS
Normal	**998.931**	1.367	2.336	7.953
Botnet	11.823	**769.107**	188.302	41.355
PortScan	65.993	18.038	**897.396**	29.160
DDoS	2.308	3	298	**1.007.978**

Tabela 7. Matriz de confusão do algoritmo Árvore Adaptiva Hoeffding, utilizando classificação em lotes e 10 características principais.

Árvore Adaptiva Hoeffding	Normal	Botnet	PortScan	DDoS
Normal	**1.005.179**	624	1.063	3.721
Botnet	5.935	**899.495**	87.822	17.335
PortScan	28.330	8.814	**961.046**	12.397
DDoS	1.362	0	128	**1.009.097**

Tabela 8. Matriz de confusão do algoritmo Bayesiano Simples, utilizando classificação em lotes e 10 características principais.

Bayesiano Simples	Normal	Botnet	PortScan	DDoS
Normal	**659.125**	28.337	101.163	221.962
Botnet	198.308	**511.462**	80.883	219.934
PortScan	63.108	10.869	**877.709**	58.901
DDoS	1.902	7	458	**1.008.220**

Tabela 9. Matriz de confusão do algoritmo Bayesiano Simples, utilizando classificação em lotes e todas as características.

Bayesiano Simples	Normal	Botnet	PortScan	DDoS
Normal	**852.183**	12.049	48.338	98.017
Botnet	87.692	**783.629**	33.874	105.392
PortScan	31.936	4.408	**946.401**	27.842
DDoS	1.098	2	196	**1.009.291**

Observando os resultados obtidos, o processamento em fluxo reduziu em até 54,55% a quantidade de falsos-positivos e falsos-negativos para o conjunto de dados utilizando 10 características e em até 66,61% quando utilizando todas as características do conjunto de dados. Este ganho em acurácia pelo processamento em fluxos é resultado da adaptação em tempo real do algorítimo, onde cada dado é primeiro classificado e depois integrado ao conjunto de dados para auxiliar na classificação de dados futuros.

7. Conclusão

Este artigo apresenta um sistema de detecção de intrusão *online* para detecção de ameaças distribuídas de rede utilizando o processamento em fluxos. São utilizados três algoritmos de aprendizado de máquina, tanto no modo de processamento em lotes como no processamento em fluxos. A base de dados é avaliada utilizando todas as características

extraídas, assim como executando uma redução no espaço para apenas 10 características.

Considerando todas as combinações de algoritmos de classificação e número de características, o processamento em fluxos apresentou uma melhora na acurácia da classificação de ameaças em relação ao processamento em lotes. Para o algoritmo bayesiano simples, o método proposto melhorou em até 17,50% a taxa de acerto e reduziu em até 66,61% a taxa de falsos-positivos e falsos-negativos. Em alguns cenários, a redução do número de características demostrou não ser tão eficiente na melhora da acurácia, porém reduziu o tempo de classificação em até seis vezes, comparado com o tempo de classificação utilizando todas as características. Considerando a grande quantidade de dados, a necessidade de uma resposta imediata e modelos de previsão compactos com baixo consumo de memória, a classificação de ameaças *online* requer técnicas de mineração de fluxo de dados para resolver as necessidades e problemas de detecção de intrusão em redes de computadores. Como trabalhos futuros, pretende-se estudar e analisar outros métodos de classificação, testar novos conjuntos de características e outras base de dados.

Referências

Bifet, A. and Gavaldà, R. (2009). Adaptive learning from evolving data streams. In *Proceedings of the 8th International Symposium on Intelligent Data Analysis: Advances in Intelligent Data Analysis VIII*, IDA '09, pages 249–260, Berlin, Heidelberg. Springer-Verlag.

Bifet, A., Holmes, G., Pfahringer, B., and Frank, E. (2010). Fast perceptron decision tree learning from evolving data streams. In *Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II*, PAKDD'10, pages 299–310, Berlin, Heidelberg. Springer-Verlag.

Bifet, A., Holmes, G., Pfahringer, B., Read, J., Kranen, P., Kremer, H., Jansen, T., and Seidl, T. (2011). Moa: A real-time analytics open source framework. In *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part III*, ECML PKDD'11, pages 617–620, Berlin, Heidelberg. Springer-Verlag.

Buczak, A. L. and Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys Tutorials*, 18(2):1153–1176.

Chen, F., Deng, P., Wan, J., Zhang, D., Vasilakos, A. V., and Rong, X. (2015). Data mining for the internet of things: Literature review and challenges. *International Journal of Distributed Sensor Networks*, 11(8):431047.

CICFlowMeter (2017). Cicflowmeter - a network traffic biflow generator and analyzer. Acessado em: 15-08-2018.

Corrêa, D. G., Enembreck, F., and Silla, C. N. (2017). An investigation of the hoeffding adaptive tree for the problem of network intrusion detection. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4065–4072.

Desale, K. S., Kumathekar, C. N., and Chavan, A. P. (2015). Efficient intrusion detection system using stream data mining classification technique. In *2015 International Conference on Computing Communication Control and Automation*, pages 469–473.

Gama, J., Sebastião, R., and Rodrigues, P. P. (2009). Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 329–338, New York, NY, USA. ACM.

Gama, J., Sebastião, R., and Rodrigues, P. P. (2013). On evaluating stream learning algorithms. *Mach. Learn.*, 90(3):317–346.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37.

Gartner (2016). Gartner says by 2020, more than half of major new business processes and systems will incorporate some element of the internet of things. Acessado em: 07-11-2018.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.

Kezih, M. and Taibi, M. (2013). Evaluation effectiveness of intrusion detection system with reduced dimension using data mining classification tools. In *2nd International Conference on Systems and Computer Science*, pages 205–209.

Kholghi, M., Hassanzadeh, H., and Keyvanpour, M. (2010). Classification and evaluation of data mining techniques for data stream requirements. In *2010 International Symposium on Computer, Communication, Control and Automation (3CA)*, volume 1, pages 474–478.

Labs, F. (2017). The hunt for iot: The rise of thingbots. Acessado em: 07-11-2018.

Lashkari, A. H., Zang, Y., Owhuo, G., Mamun, M. S. I., and Gil, G. D. (2017). Cicflowmeter - a network traffic biflow generator and analyzer. Acessado em: 15-08-2018.

Lobato, A. G. P., Andreoni, M., and Duarte, O. C. M. B. (2016). Um sistema acurado de detecção de ameaças em tempo real por processamento de fluxos.

Pecht, M. G. and Kang, M. (2019). *Machine Learning: Fundamentals*. IEEE.

Schuartz, F. C., Fonseca, M., and Munaretto, A. (2017). Sistema distribuído para detecção de ameaças em tempo real utilizando big data. In *XXXV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais - SBrT 2017*.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2017). Cicids2017. Acessado em: 15-08-2018.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,*, pages 108–116. INSTICC, SciTePress.

Sun, J. (2010). Application of data mining for decision tree model of multi-variety discrete production and manufacture. In *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*, pages 724–728.

Symantec (2018). Internet security threat report, volume 23. Acessado em: 07-11-2018.

Seleção *Online* de *Features* em *Streaming* Baseada em Alpha-Investing$^+$ para Dados de Ataques DDoS

Alissar Ali Moussa[1], **Michele Nogueira**[1], **André Luiz P. Guedes**[1]

[1]Centro de Ciência de Segurança Computacional (CCSC)
Universidade Federal do Paraná (UFPR)

{aamoussa,michele,andre}@inf.ufpr.br

Abstract. *Distributed Denial of Service (DDoS) attacks are a significant security threat, because they are financially harmful to companies and organizations as they affect services' availability. As attacks become more complex, traditional intrusion detection systems are limited to attacks in advanced stage or they can only raise an alarm when target is already damaged. However, the constant search for efficient detection systems includes selecting the most relevant features, i.e., the characteristics that will be used for detection. The more associated to the attacks' operation mode, the more relevant is the feature. Nevertheless, as attacks get more soffisticated, it is important to consider an online approach to feature selection. Thus, the feature selection algorithm must be able to work with streaming data and features. In this work, we present a feature selection method for streaming DDoS attacks data. The method is based on the* $alpha$*-investing algorithm and we name it* $alpha$*-investing*$^+$*, which is able to sequentially add and test features. CTU-13 botnet dataset and CICIDS2017 attacks dataset are used to verify the efficiency of our method. A qualitative analysis is present and it suggests that this technique can rapidly select features in an ongoing attack.*

Resumo. *O ataque de negação de serviço distribuído (DDoS) é uma ameaça significativa de segurança, pois compromete financeiramente empresas e organizações ao afetar a disponibilidade de seus serviços. Com a sofisticação das técnicas utilizadas para gerar esses ataques, os sistemas de detecção tradicionais se limitam a ataques DDoS em estágios avançados ou quando o alvo está comprometido. Entretanto, a busca por maior eficiência nesses sistemas de detecção é uma constante e passa pelo problema de selecionar as features mais relevantes, ou seja, as características a serem utilizadas como base para a detecção. Quanto mais associada com o modo de operação do ataque, mais relevante é a feature. No contexto de ataques DDoS, é interessante que a seleção de features consiga funcionar sob dados e/ou features em streaming. Desta forma, apresentamos um método para seleção de features em streaming na detecção de ataques DDoS. Ele está embasado na nossa adaptação do algoritmo* α*-investing, chamado de* α*-investing*$^+$*, o qual atua de forma sequencial na adição e nos testes das features. O método é aplicado nas bases de dados CTU-13 e CICIDS2017 e através de uma análise qualitativa, sugere-se que a técnica pode selecionar rapidamente features relevantes de um ataque em curso.*

1. Introdução

Um ataque volumétrico de negação de serviço distribuído (DDoS - *Distributed Denial of Service*) ocorre tradicionalmente quando vários dispositivos finais infectados (*a.k.a.*, bots)

realizam ataques coordenados por uma máquina mestre contra um alvo a fim de afetar a disponibilidade de seus recursos. Esse tipo de ataque representa uma ameaça significativa às empresas e às organizações, pois estas são afetadas financeiramente. Além do impacto financeiro, os ataques DDoS também podem ter motivações políticas, como foi o caso do ataque DDoS executado pelos ciberativistas da Anonymous contra o Banco Original em maio de 2017, o qual tornou indisponível o site da instituição. No primeiro trimestre de 2018, houve 1,4 milhão de ataques DDoS no mundo inteiro, com duração média de 14 horas e pico de tráfego de dados registrado em 153 Gbps [Borini 2018].

Devido à sofisticação das técnicas utilizadas para gerar ataques DDoS, as ferramentas tradicionais de detecção só conseguem reportá-los quando atingem estágios avançados e o alvo já está comprometido, sendo tarde demais para uma ação eficiente contra esses ataques, conforme demonstrado através do modelo em [Santos et al. 2017]. Assim, é cada vez mais necessário que as técnicas de detecção sejam capazes de identificar comportamentos suspeitos antes da sobrecarga do alvo, para que um ataque em potencial possa ser impedido ou que seja possível mitigar eficientemente os seus efeitos. Além disso, os ataques DDoS volumétricos estão relacionados a um volume de dados muito grande, com alta dimensionalidade, que dificulta a análise em tempo real. A Maldição da Dimensionalidade, descrita por Bellman em 1957, é um problema causado pelo aumento exponencial de volume associado com a adição de dimensões extras em um espaço Euclidiano [Bellman 1966]. Um dos meios empregados com frequência para solucionar ou mitigar esse problema, tratando-se de dados reais, é a seleção de *features*. *Feature* é sinônimo de dado de entrada ou atributo [Guyon and Elisseeff 2008]. Em um conjunto de dados de diagnósticos médicos, por exemplo, as *features* podem ser os sintomas de cada doença, isto é, o conjunto de atributos que descrevem uma determinada enfermidade. Em nosso trabalho, as *features* são os atributos presentes em dados de captura de pacotes, tais como endereço IP de origem, protocolo, porta de origem, entre outros.

Tradicionalmente, os algoritmos de seleção de *features* consideram que o espaço de dados e de *features* já são conhecidos [Li et al. 2017]. Porém, tratando-se de aplicações do mundo real nem sempre os conjuntos completos de dados e/ou de *features* estão disponíveis, impossibilitando o uso das técnicas tradicionais. Assim, é preciso que um algoritmo de seleção de *features* para dados dessa natureza consiga executar a tarefa de seleção sem conhecer o espaço total de *features* ou de dados [Hu et al. 2018], o que é um grande desafio e um relevante problema de pesquisa. Nos últimos anos, o interesse pela seleção de *features* em *streaming* tem aumentado devido ao crescimento das aplicações de *big data* e que geram grande volume de dados. Diversos trabalhos propõem técnicas de seleção de *features* em *streaming* fundamentados em conceitos como a regularização de esparcidade [Jialei Wang et al. 2014] e a probabilidade condicional [Wu et al. 2013]. Entretanto, as publicações referentes aos cenários de ataques DDoS em sua maioria abordam o problema da seleção de *features* de maneira *offline*, utilizando técnicas tradicionais, como os métodos de filtro [Osanaiye et al. 2016] e de *clustering* [Idhammad et al. 2018].

Neste trabalho, apresentamos um método de detecção online de *features* para dar suporte à detecção ou predição de ataques DDoS. Este método está fundamentado na nossa adaptação do algoritmo α-investing [Zhou et al. 2005], chamada de α-investing$^+$, para seleção de *features* em *streaming*. O método é *online* e constrói um conjunto de *features* selecionadas no momento de sua descoberta, após uma avaliação de independência

entre a *feature* recém-descoberta e as já existentes no conjunto. Cada variável candidata a *feature* selecionada é examinada apenas uma vez, e a verificação constante de independência entre as variáveis reduz a possibilidade de haver *features* redundantes no conjunto selecionado. O algoritmo foi adaptado para lidar com dados de ataques de DDoS, que são não-paramétricos e categóricos, sugerindo a necessidade de testes que atendam a essas características. Por isso, foi utilizado o Teste de Qui-Quadrado para Independência entre as variáveis com o valor-p estimado a partir de uma simulação de Monte Carlo.

O método proposto foi avaliado sob as bases de dados CTU-13 [García et al. 2014] e CICIDS2017 [Sharafaldin et al. 2018]. A primeira contém dados de tráfego de *botnets* testadas na Czech Technical University (CTU). Os dados estão divididos em 13 capturas, referidas pelos autores como cenários, e neste trabalho são utilizados os cenários cujos bots se coordenam e geram ataques DDoS. Já a segunda base foi construída a partir de uma série de diversos ataques, em um cenário experimental criado na University of New Brunswick, e entre esses ataques está um DDoS. Uma análise qualitativa é apresentada, confrontando os resultados apresentados pelo método com o conhecimento dos autores sobre a base. Os resultados sugerem que o método consegue selecionar as *features* de maneira rápida, mas que pode apresentar resultados inconclusivos dependendo do cenário ao qual o algoritmo é submetido. Em um cenário mais diverso, com tráfego de dados variando entre períodos com mais ou menos volume de dados, as *features* selecionadas são mais fieis ao contexto do ataque analisado.

Este trabalho está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados e o estado da arte. A Seção 3 descreve o método proposto fundamentado no algoritmo α-investing adaptado para dados de ataques DDoS. A Seção 4 expõe o método, a sua implementação e discute os resultados obtidos. A Seção 5 conclui o trabalho.

2. Trabalhos relacionados

Os métodos tradicionais de seleção de *feature* podem ser classificados em três categorias: filtro, *wrapper* e *embedded* [Guyon and Elisseeff 2003]. As técnicas de filtro selecionam subconjuntos de *features* na etapa de pré-processamento, sem auxílio de um classificador. A avaliação para cada subconjunto envolve diversas métricas como ganho de informação, consistência e relevância [Eskandari and Javidi 2016]. Os métodos *wrapper* utilizam classificadores para avaliar os subconjuntos de *features* gerados. Por fim, a técnica *embedded* atrela o processo de seleção de *feature* ao treinamento do modelo.

Como os métodos citados acima consideram que ambos os espaços de *feature* e de dados já são conhecidos previamente, há a necessidade de técnicas que consigam tratar dados gerados em tempo real (*online*) e de maneira contínua. A literatura menciona técnicas *online* seleção de *features* referentes a dados em streaming e a *features* em *streaming*. No primeiro caso, considera-se que o conjunto de *features* é conhecido previamente e o espaço de dados é atualizado com a chegada de novas entradas. Já no segundo caso, o tamanho do conjunto de *features* é desconhecido e potencialmente infinito [Li et al. 2017]. Neste trabalho, o foco será nos algoritmos que tratam de *features* em *streaming*.

Em [Perkins et al. 2003] é proposta uma técnica de seleção incremental de *features* chamada *Grafting* que, embora não pertença a mesma categoria dos algoritmos discutidos neste trabalho, é precursora dos métodos que lidam com dados e *features* em

streaming. Segundo os autores, o *Grafting* constrói gradativamente do conjunto de *features* enquanto treina um modelo de predição usando gradiente descendente. Contudo, essa técnica necessita que um parâmetro de regularização λ seja definido previamente para determinar quais *features* serão selecionadas a cada iteração. Por consequência, requere-se um conhecimento prévio de todo o espaço de *features*, impossibilitando o uso desse método para selecionar *features* em *streaming*.

[Wu et al. 2013] apresentam o *Fast-OSFS*, método que analisa a relevância das *features* e remove *features* redundantes dentre as já selecionadas. Baseando-se no conceito de probabilidade condicional, o algoritmo determina se duas *features* x_1 e x_2 têm forte relevância, fraca relevância ou irrelevância entre si. Seus resultados são comparados com o α-investing, técnica na qual se baseia o método descrito neste trabalho. Os autores argumentam que o α-investing somente tem desempenho melhor que o *Fast-OSFS* quando existe conhecimento sobre a estrutura do espaço de *features*. Além disso, nos resultados apresentados não estão inclusas bases com dados de ataques DDoS.

[Eskandari and Javidi 2016] afirmam que as *features* em *streaming* são partes fundamentais da maiorias das aplicações de mundo real. Além disso, apontam que o uso de um algoritmo de seleção de *features* em *streaming* pode ser utilizado em dois casos. O primeiro consiste no cenário em que não se conhece o espaço de *features*, e o segundo caso se refere ao cenário quando se conhece o espaço de *features*, mas a seleção pode melhorar o desempenho e a acurácia na classificação. Eles propõem um algoritmo fundamentado no conceito de conjuntos aproximados, que expressam a imprecisão em regiões limítrofes de um conjunto. As *features* são selecionadas a partir de uma análise de significância.

Os trabalhos recentes que abordam especificamente os ataques DDoS tendem a propor soluções que se aproximam das técnicas tradicionais, que é caso de [Osanaiye et al. 2016] e [Idhammad et al. 2018]. [Osanaiye et al. 2016] propõem um método que baseado em filtro para seleção de *features* em dados de ataques DDoS em nuvem. Nesse trabalho, quatro filtros são combinados (*Information Gain*, *Gain Ratio*, Qui-Quadrado e *ReliefF*) para obter uma seleção ótima de atributos em dados históricos de ataques DDoS. Para que um filtro consiga retornar um vetor de características, ou seja, o conjunto de *features* selecionadas, é necessário que o conjunto de dados de entrada seja rotulado. Isso impede a análise de outros dados que não tenham rótulo associado, como é o caso das bases utilizadas em nosso trabalho. Estas foram geradas a partir de ferramentas de monitoramento de tráfego, contendo informações sobre os pacotes da rede, sem qualquer tipo de distinção entre dados que são parte de um ataque e entre os que não são.

A proposta presente no trabalho de [Idhammad et al. 2018] segue uma abordagem semi-supervisionada, onde é aplicado um algoritmo de *clustering* para reduzir o espaço de *features* e encontrar tráfego anômalo. Assume-se que são produzidos três *clusters*, um que contém apenas tráfego normal, outro apenas com tráfego de ataque DDoS e um último com dados de ataques DDoS e tráfego normal. Estes *clusters* são gerados a partir de *NetFlows* oriundos de um histórico e são utilizados na classificação de ataques DDoS. Assim como [Osanaiye et al. 2016], o método de [Idhammad et al. 2018] é *offline*, ou seja, não é em tempo real, e não lida com dados em *streaming*, portanto não são aplicáveis em situações onde a análise tenha que ser executada sob demanda.

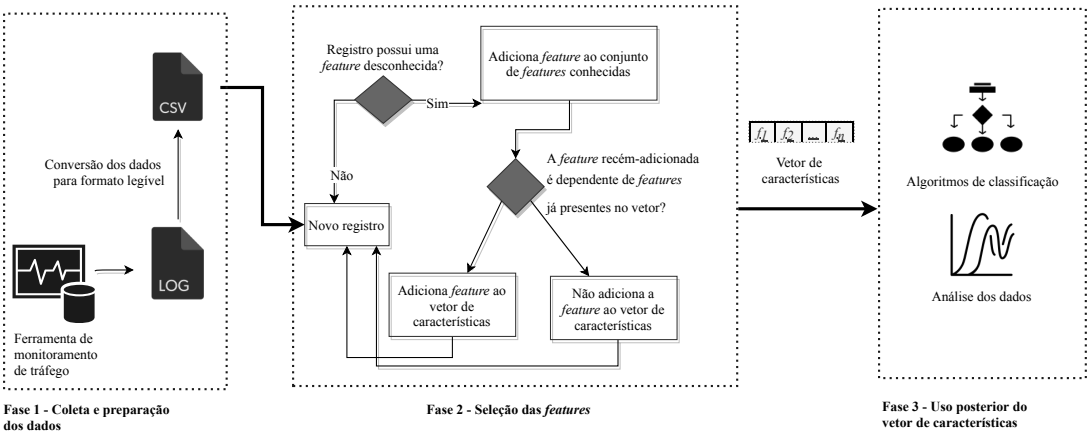


Figura 1. Visão geral das fases do método proposto.

3. Método de seleção de *features* em *streaming* para detecção de ataques

Nesta seção apresentamos o método de seleção de *features* em streaming para detecção de ataques volumétricos DDoS. Como ilustra a Figura 1, nosso método é composto por três fases: tratamento dos dados, análise dos dados pela versão adaptada do algoritmo α-*investing* e uso das *features* selecionadas. A primeira fase consiste em coletar e processar os dados em tempo real, a fim de prepará-los para a análise. A segunda fase aplica a seleção de *features* nos dados coletados até um dado momento. A terceira fase é responsável por obter um vetor de características gerado a partir da seleção executada na segunda fase, destinando-o a algoritmos de classificação ou outras atividades.

3.1. Tratamento dos dados

Na primeira etapa, assume-se que alguma ferramenta de captura e monitoramento de tráfego, como o tcpdump, Wireshark ou Netflow, estejam monitorando e coletando dados do tráfego da rede. Nesses dados, disponíveis em *logs*, estão o conteúdo de cada pacote ou fluxo de rede capturado e estes passam por um processo de *parsing*, para obter uma *string* ou um conjunto de *strings* em um formato legível para alimentar a segunda fase. A fase de tratamento de dados pode estar integrada à segunda fase se cada entrada no *log* for analisada separadamente, considerando a situação de uma análise *online*, feita sob demanda. Nesse caso, o algoritmo é alimentado com um registro do *log* por vez, que é o processo apresentado neste trabalho. Um registro é uma entrada no *log*, que contém as informações sobre o pacote capturado na ferramenta de análise de tráfego. Outra possibilidade de entrada para o algoritmo é um conjunto de registros, para que se tenha um pseudo-histórico de dados. Essa situação seria possível se for observado algum comportamento estranho no *log* que deva ser considerado para a análise aplicada aos dados conseguintes.

3.2. Seleção de *features* pelo algoritmo α-*investing*$^+$

O α-*investing* é um algoritmo de seleção de *features* em *streaming*, que visa diminuir a taxa de falsas descobertas (FDR - *False Discovery Rate*) ao ajustar dinamicamente um limiar α_i. Este limiar determina se uma *feature* recém-descoberta pode ser adicionada ao conjunto de *features* selecionadas [Zhou et al. 2005]. Neste trabalho, seguimos o algoritmo α-*investing*$^+$, uma adaptação nossa do algoritmo α-*investing* original, que não

Algoritmo 1: Algoritmo α-investing$^+$

Dados: Logs de rede
Resultado: Conjunto de *features* selecionadas

1 $W \leftarrow [0.5]$; `// Conjunto de pesos`
2 $w_0 \leftarrow W_0$; `// Peso inicial`
3 $F \leftarrow \{\}$; `// Conjunto de` *`features`* `detectadas`
4 $S \leftarrow \{\}$; `// Conjunto de` *`features`* `selecionadas`
5 $\alpha_\Delta \leftarrow 0.5$;
6 i $\leftarrow$ 1;
7 **enquanto** *novas features forem detectadas* **faça**
8 adicionarFeature(f_i, F);
9 $\alpha_i \leftarrow w_i/2 * i$;
10 **para** *cada* $s \in S$ *tal que* $s \neq f_i$ **faça**
11 tabela $\leftarrow$ gerarTabelaContingencia(f_i, m);
12 resultado $\leftarrow$ chiQuadrado(tabela);
13 **se** *resultado[valorP]* $\leq \alpha_i$ **então**
14 $w_{i+1} \leftarrow w_i - \alpha_i$;
15 **senão**
16 adicionarFeature(f_i, S);
17 $w_{i+1} \leftarrow w_i + \alpha_\Delta - \alpha_i$;
18 **fim**
19 **fim**
20 i $\leftarrow$ i + 1;
21 **fim**

requer um tamanho fixo para o conjunto de *features*, admitindo a adição de novas *features* geradas dinamicamente ao longo do tempo ou a partir de outras já existentes. Além disso, este algoritmo funciona independentemente de como as *features* são geradas. O algoritmo α-*investing*$^+$ foi criado para se adequar à natureza dos dados tratados diante de cenários de ataques DDoS volumétricos, sendo estes não-paramétricos e categóricos. O α-*investing*$^+$ é apresentado em detalhes a seguir e seus passos são ilustrados no Algoritmo 1.

Seja $D = \{d_1, d_2, ..., d_n\}$ um conjunto de dados gerados até um momento t, tal que $n > 0$ e cada $d_i \in D$, sendo $0 < i \leq n$ e d_i uma j-tupla $< f_1, ..., f_j >$ (também chamado de **registro** neste artigo). Temos que para $j > 0$, f_k é um campo de texto que descreve uma característica de um d_i e $0 < k \leq j$. Cada f_j é denotado como ***feature***. Sejam $F = \{f_l$: uma feature detectada $| \; l > 0\}$ e $S = \{s_m$: uma feature selecionada $| \; m > 0\}$. Ambos os conjuntos iniciam-se vazios e não admitem elementos repetidos. É importante notar que cada *feature* única presente nos registros do conjunto D são pertencentes ao conjunto F e $S \subseteq F$. O algoritmo, ao ser iniciado, recebe um dado registro d_i (linha 7 em Algoritmo 1), que também pode ser representado como d_{t_s}, tal que t_s é o momento inicial da análise e $t_s > t$. Para que uma *feature* seja adicionada ao conjunto F (linha 8 em Algoritmo 1), são executados os seguintes passos:

1. Lê-se um novo registro;
2. Verificam-se quais *features* têm seus valores preenchidos naquele registro;
3. Descartam-se as *features* com valores não-preenchidos;

4. Para cada *feature* que não foi descartada, é verificado se ela já pertence a F;
   (a) Se a *feature* está em F, a execução é interrompida e o passo 4 é executado para a próxima *feature*;
   (b) Se *feature* não está em F, ela é adicionada ao conjunto.

Uma *feature* f_i é adicionada a S se ela não tiver relação de dependência com algum s_m em S (linha 16 em Algoritmo 1). O limiar α_i, atualizado na linha 9 do Algoritmo 1, determina a probabilidade de se incluir uma *feature* irrelevante ao modelo ao i-ésimo passo do algoritmo. O peso w_i representa o número de possíveis falsos positivos a serem incluídos ao modelo nos passos seguintes. As linhas 14 e 17 do Algoritmo 1 mostram a atualização de w_i quando f_i é dependente de algum elemento em S ou independente de algum elemento em S, respectivamente. Um falso positivo, no contexto deste trabalho, é uma *feature* $x \in S$ tal que existe pelo menos um $y \in S$ que possui uma relação de dependência com x e $x \neq y$. α_i é ajustado a partir do peso w_i. O valor-p corresponde à probabilidade das *features* f_1 e f_2 serem variáveis independentes e pode ser obtido a partir de um teste estatístico.

Como mencionado anteriormente, os dados de ataques DDoS são, frequentemente, não-paramétricos e categóricos. Assim, é necessário que o teste estatístico utilizado seja adequado a esse tipo de dados. Por isso, neste trabalho optou-se por utilizar o Teste do Chi-Quadrado para Independência de Variáveis, que basicamente é um teste de hipóteses. Neste trabalho, o valor-p é obtido a partir de uma simulação de Monte Carlo utilizada para o Teste Chi-Quadrado para Independência de Variáveis (linha 12 em Algoritmo 1).

O teste Chi-Quadrado pode ser calculado a partir de uma tabela de contingência que determina a frequência dos valores observados entre as categorias de duas variáveis v_1 e v_2 [Freedman et al. 1998]. Cada célula nessa tabela aponta a frequência que cada um dos valores de v_1 ocorre com cada um dos valores de v_2 em uma mesma observação. Em relação às hipóteses que serão testadas, a hipótese nula, denotada como H_0, e a hipótese alternativa, denotada como H_1, são definidas da seguinte forma:

$$H_0\text{: } v_1 \text{ e } v_2 \text{ são independentes entre si}$$

$$H_1\text{: } v_1 \text{ e } v_2 \text{ são dependentes entre si}$$

Tratando-se de dados de rede, uma tabela de contingência pode facilmente tornar-se esparsa, devido à grande variedade de informações contidas em cada variável categórica. Quando o teste Chi-Quadrado é aplicado em tabelas que contém frequências observadas menores do que 1, os resultados retornados não são confiáveis, incluindo o valor-p. Nesse caso, uma opção viável é estimar o valor-p através de uma simulação de Monte Carlo [Kim et al. 2006]. A partir de uma simulação de Monte Carlo é possível obter um valor empírico do erro de Tipo I, que ocorre quando H_0 é rejeitada quando na realidade ela é verdadeira. O teste é replicado diversas vezes assumindo a H_0 como verdadeira e o erro Tipo I empírico é a amostra proporcional de estatísticas significantes entre os testes replicados [Rizzo 2007].

O critério de parada do α-*investing*$^+$ é arbitrário, já que enquanto o algoritmo estiver sendo executado, serão esperadas detecções de novas *features*. Ao fim da execução, o conjunto S é denominado vetor de características e assume-se que estas características são as que melhor representam o conjunto de dados analisados do momento t_s até a parada.

3.3. Uso posterior das *features* selecionadas

A fase dois retorna um vetor de características que pode ser aplicado para pré-processamento de dados volumétricos, a fim de reduzir as dimensões e compactar o tamanho da base. Além disso, o vetor pode ser utilizado para extrair representações de bases de ataques DDoS que são usadas em classificadores ou em algoritmos de *clustering* para distinguir os diferentes tipos de tráfego. As características também podem ser aplicadas na gerência de segurança, para que saiba quais variáveis monitorar quando o objetivo é identificar um ataque, assim como para ajustar sistemas de detecção de ataques. Neste último, a seleção de *features* pode ser útil para refinar os padrões aos quais os dados são testados para que se verifique se há um ataque em curso, como é o caso das ferramentas de detecção de intrusão.

4. Metodologia

Esta seção detalha a implementação da Fase 1 e Fase 2 do método de seleção de *features* em *streaming* apresentado neste trabalho, assim como os resultados obtidos. Para isso, são utilizadas duas bases de dados públicas: CTU-13 e CICIDS2017, que contém dados de tráfego normal e de tráfego de ataques DDoS. Em ambas as bases são registrados diferentes tipos de ataques DDoS, proporcionando uma diversificação da análise e dos resultados apresentados.

4.1. Preparação dos dados

As duas bases são disponibilizadas no formato pcap, que contém a captura de cada pacote da rede durante o período de monitoramento. Para que a análise fosse facilitada, em termos de diminuição do tamanho dos arquivos a serem processados e também de legibilidade das informações, cada arquivo foi convertido para o formato CSV.

4.2. Detalhes técnicos da implementação do α-*investing*$^+$

O algoritmo foi implementado nas linguagens Python[1] na versão 3.7 e R na versão 3.5, com auxílio das bibliotecas Pandas[2] e rpy2[3], ambas disponíveis para a linguagem Python. Pandas é uma biblioteca que oferece ferramentas para análise de dados, estatística e gráficos. rpy2 é uma biblioteca que realiza a interface entre o Python e o R. A princípio, apenas o Python estava sendo utilizado, porém o Pandas não possui uma função que calcula o valor-p a partir de uma simulação de Monte Carlo para o teste do Chi-Quadrado. Por isso, optou-se por executar uma instância do R a partir do *script* escrito em Python para fazer a chamada da função do R `chisq.test()`, que atende aos requisitos para os dados utilizados neste trabalho.

4.3. CTU-13

CTU-13 é uma base pública de dados de tráfego de *botnets*, capturada pela CTU University na República Tcheca, em 2011. A topologia empregada para criar a base consiste em um conjunto de máquinas virtuais criadas sob um *host* Debian, sendo que as

[1]Documentação da versão 3.7: `https://docs.python.org/3.7/`

[2]Documentação da biblioteca Pandas: `http://pandas.pydata.org/pandas-docs/stable/`

[3]Documentação da biblioteca Pandas: `https://rpy2.readthedocs.io/en/version_2.8.x/index.html`

virtualizações executam Windows XP SP2. A base foi composta a partir da execução de 13 *malwares* e contém dados de tráfegos normais, de *botnet* e de *background*. Para cada *malware*, [García et al. 2014] descrevem que um *cenário de botnet* é a infecção das máquinas virtuais por um *malware* específico. Entre as características interessantes dessa base, destacam-se o fato de que nela estão descritos apenas ataques reais de *botnets* de diferentes tipos e que há tráfego desconhecido/não-classificado obtido de uma rede de grande porte.

Cada cenário foi projetado para representar o comportamento de algum *malware*. O tráfego de dados capturado pela ferramenta `tcmpdump`, que gerou arquivos pcap e *NetFlow* para cada cenário. Neste trabalho, serão utilizados apenas os arquivos pcap, por conterem as informações sobre os pacotes capturados individualmente e por os dados não serem rotulados. É importante notar que os arquivos pcap disponibilizados pelos autores da base contém dados somente do tráfego de *botnets*, excluindo os tráfegos normais e de *background* para preservar a privacidade da rede. Para emular uma execução em tempo real, em cada experimento é fornecido apenas um pacote por vez para análise, dentro do período estipulado.

Dos 13 cenários criados, três deles são referentes a ataques DDoS volumétricos: cenários 4, 9 e 10. No cenário 4 houve um ataque DDoS de ICMP e UDP *Flood*, teve duração maior que 4 horas, o volume de dados capturado foi de 55 GB. No cenário 9 foi executado um ataque DDoS de UDP *Flood*, com duração de pouco mais de 5 horas, gerando um volume de dados de 94 GB, o maior entre os três cenários. Por fim, o cenário 10 descreve um ataque DDoS de ICMP *Flood*, com duração de quase 5 horas e com volume de dados de 73GB. Neste trabalho, foi utilizado o cenário 4, por se tratar de uma captura envolvendo ataque DDoS com *flooding* de dois protocolos distintos.

A topologia de rede construída para realizar as capturas de cada cenário tinham controle de largura de banda e o tráfego de saída para Internet não estava sendo filtrado, a fim de que pudessem a rede pudesse receber ataques reais. Tanto o tráfego normal e de *background* quando o tráfego de ataques DDoS estavam sendo capturados pela mesma ferramenta. No cenário 4, as *botnets* utilizaram protocolo IRC e, como já descrito antes, executaram ataques DDoS de ICMP e UDP *Flood*.

4.3.1. Experimentos no Cenário 4

Nesse cenário, foram executados três experimentos. O experimento 1 compreende o período das 10:32 até às 10:52, ao qual foi registrado um intenso fluxo de dados. O experimento 2 abrange o início do *log* até o início do período utilizado no experimento 2. Finalmente, no experimento 3, o algoritmo foi executado sobre o *log* completo. O intuito é analisar o comportamento do algoritmo em três situações: durante o momento mais crítico do ataque, o momento que antecede o período mais crítico e o cenário completo, pré e pós-período mais crítico.

As *features* selecionadas em cada experimento são apresentadas na Tabela 1. A Tabela 2 mostra as *features* que apresentaram relação de dependência em cada experimento e, portanto, não foram selecionadas. Em todos os experimentos, foram detectadas 18 *features*, sendo elas: [‘SourceAddr’, ‘SourcePort’, ‘DestAddr’, ‘Protocol’, ‘Length’,

	Features selecionadas
Experimento 1	[‘SourceAddr’, ‘Protocol’, ‘DestPort’, ‘TcpAck’, ‘TcpFlagAck’, ‘TcpFlagRst’]
Experimento 2	[‘SourceAddr’, ‘Protocol’, ‘Length’, ‘FrameLen’, ‘IpTtl’, ‘IpOffset’, ‘Severity’]
Experimento 3	[‘SourceAddr’, ‘Protocol’, ‘Length’, ‘FrameLen’, ‘IpTtl’, ‘IpOffset’, ‘Severity’]

Tabela 1. Resultados dos experimentos no cenário 4

	Pares de features que são dependentes entre si
Experimento 1	[(’Length’, ’SourceAddr’), (’FrameLen’, ’SourceAddr’), (’IpTtl’, ’SourceAddr’), (’IpOffset’, ’Protocol’), (’TcpFlagPsh’, ’SourceAddr’), (’TcpFlagSyn’, ’SourceAddr’), (’TcpFlagFin’, ’SourceAddr’), (’TcpStream’, ’SourceAddr’), (’TcpWindowSize’, ’SourceAddr’), (’Severity’, ’SourceAddr’)]
Experimento 2	[(’DestPort’, ’Protocol’), (’TcpAck’, ’Length’), (’TcpFlagAck’, ’Length’), (’TcpFlagRst’, ’SourceAddr’), (’TcpFlagPsh’, ’SourceAddr’), (’TcpFlagSyn’, ’SourceAddr’), (’TcpFlagFin’, ’SourceAddr’), (’TcpStream’, ’Length’), (’TcpWindowSize’, ’SourceAddr’)]
Experimento 3	[(’DestPort’, ’Protocol’), (’TcpAck’, ’Length’), (’TcpFlagAck’, ’Length’), (’TcpFlagRst’, ’SourceAddr’), (’TcpFlagPsh’, ’SourceAddr’), (’TcpFlagSyn’, ’SourceAddr’), (’TcpFlagFin’, ’SourceAddr’), (’TcpStream’, ’Length’), (’TcpWindowSize’, ’SourceAddr’)]

Tabela 2. Dependência entre as *features* observadas no cenário 4

‘FrameLen’, ‘DestPort’, ‘IpTtl’, ‘IpOffset’, ‘Severity’, ‘TcpAck’, ‘TcpFlagAck’, ‘TcpFlagRst’, ‘TcpFlagPsh’, ‘TcpFlagSyn’, ‘TcpFlagFin’, ‘TcpStream’, ‘TcpWindowSize’].

No **experimento 1**, é interessante observar que todas as *features* que não foram selecionadas têm relação de dependência com a *feature* SourceAddr (endereço IP de origem) (Tabela 2). Dentre essas relações, destaca-se a *feature* Length (tamanho do pacote), que nos outros experimentos foi dada como independente de todas as *features* já existentes no conjunto M. Esse resultado sugere que durante o pico deste ataque cada *host* atacante estava enviando muitos pacotes do mesmo tamanho ou de tamanhos parecidos, a fim de sobrecarregar a rede.

Porém, a variação do tamanho do pacote é um sinal importante de que um ataque DDoS possa estar em curso, o que nos leva a acreditar que essa *feature* deveria estar presente em um conjunto que seleciona as *features* mais representativas do tráfego de um ataque DDoS. Além disso, nota-se que três *features*, TcpAck, TcpFlagAck e TcpFlagRst, que referentes ao protocolo TCP, foram selecionadas. Considerando o fato de que a rede sofria ataques de ICMP e UDP *Flood*, essas *features* não são muito descritivas ou representativas em relação a esse tipo de ataque. Portanto, esse experimento que ana-

lisa exclusivamente o momento de tráfego mais intenso do ataque apresentou resultados inconclusivos.

O **experimento 2** e o **experimento 3** apresentaram o mesmo conjunto de *features* selecionadas e os mesmos pares de *features* que têm relação de dependência. Isso sugere que as *features* detectadas e selecionadas no estágio inicial da análise influenciam no conjunto final, já que as *features* não passam por uma reavaliação de relevância ao decorrer da análise. Entretanto, as *features* selecionadas nos dois experimentos se mostram mais relevantes ao contexto dos ataques de ICMP e UDP *Flood* se comparadas ao experimento 1. A *feature* Length está presente no conjunto, indicando que houve maior variação no tamanho do pacote em relação tanto no cenário completo quando no período pré-ataque.

Ainda referente aos experimentos 2 e 3, nota-se que foi selecionada a *feature* IpOffset, relativa à fragmentação ICMP, indicando que o tamanho máximo de um datagrama IP foi extrapolado, sendo necessária a sua divisão em pacotes menores. Esse acontecimento é comum em ataques DDoS relacionados ao protocolo ICMP e, de fato, pode ser considerado representativo no contexto desse cenário. Outra *feature* selecionada interessante é o IpTtl, que indica o tempo que o pacote é mantido pelo *host* antes de ser descartado, já que valores anormais de TTL (*Time To Live*) pode denunciar um ataque em curso [Yamada and Goto 2012]. Essa *feature* então pode ser considerada relevante para ambos os ataques de ICMP e UDP *Flood*.

4.4. CICIDS2017

O CICIDS2017[Sharafaldin et al. 2018] é uma base pública de 2017 que contém dados de ataques recentes, disponibilizada pela *University of New Brunswick*. Na base estão armazenados registros gerados por Sistemas de Detecção de Intrusão (IDS)e por Sistemas de Prevenção de Intrusão (IPS), incluindo dados de ataques de força-bruta, ataques *heartbleed*, *botnets*, ataques DoS e DDoS, ataques Web e ataques de infiltração. O período de captura iniciou-se às 09:00 de uma segunda-feira, 3 de julho de 2017 e terminou às 17:00 de uma sexta-feira, 7 de julho de 2017, durando cinco dias. Em cada dia, houve um conjunto de ataques diferentes:

- Segunda-feira: dados benignos, sem ataques;
- Terça-feira: Ataques de força-bruta sob os serviços STFP e SSH;
- Quarta-feira: Ataques DoS e de *Heartbleed*
- Quinta-feira: Ataques Web e de infiltração;
- Sexta-feira: Ataques DDoS, *botnets* e *PortScans*

Neste trabalho, serão avaliados os resultados do nosso método sob os dados capturados na sexta-feira. Nesse dia pela manhã foi executado o ataque de botnets e durante a tarde houve o ataque DDoS. O ataque DDoS foi executado através da ferramenta *Low Orbit Ion Canon* (HOIC), que enviava requisições UDP, TCP ou HTTP às vitimas. Os atacantes eram máquinas Windows 8.1 e a vítima era um servidor *web* sob o Ubuntu 16.

Nesta base foi executado um experimento apenas, sob todo o conjuntos de dados registrados durante a tarde de sexta-feira, período ao qual o ataque DDoS foi desempenhado. No total, foram capturados pouco mais de 8,3 milhões de pacotes. Foram detectadas 16 *features* neste conjunto de dados e o vetor de características foi retornado com 5 elementos, demonstrando uma redução importante no espaço de *features*. Na Tabela 3 é

Vetor de características	['DestPort', 'SourcePort', 'Protocol', 'IpFlags', 'IpLen']
Features dependentes entre si	[('SrcAddr', 'DestPort'), ('IpTtl', 'DestPort'), ('FrameLen', 'DestPort'), ('FrameProtocols', 'DestPort'), ('TcpAck', 'DestPort'), ('TcpFlags', 'DestPort'), ('TcpFlagsAck', 'DestPort'), ('TcpLen', 'DestPort'), ('TcpWindowSize', 'DestPort')]
Features detectadas	['DestPort', 'DestAddr', 'SourcePort', 'SrcAddr', 'Protocol', 'IpFlags', 'IpLen', 'IpTtl', 'FrameLen', 'FrameProtocols', 'TcpAck', 'TcpFlags', 'TcpFlagsAck', 'TcpLen', 'TcpWindowSize', 'IpFragment']

Tabela 3. Resultados dos experimento executado sob os dados observados durante o ataque DDoS na tarde de sexta-feira

possível visualizar as *features* selecionadas e detectadas, assim como as *features* que são dependentes entre si.

Nota-se que uma característica importante presente no vetor é o DestPort, já que esta *feature* é relevante nos ataques de TCP e HTTP *Flooding*. Este último ataque é especialmente mais díficil de ser detectado, já que não são utilizados pacotes mal-formados como nos outros tipos de *Flood*, atuando na camada de aplicação. O atacante pode sobrecarregar um servidor *web* ao criar múltiplas requisições GET e POST, cujos dados de monitoramento refletem na porta 80 ou 8080. Outra *feature* que se destaca é a do tamanho dos datagramas IP, que podem ser oriundos de um ataque de UDP *Flooding* e que indicam possíveis ataques em curso, em especial se o tamanho é crescente ao longo de tempo.

Um fato que chama atenção é que não há *features* no vetor de características que são exclusivamente relacionadas ao ataque de TCP *Flooding*, como aconteceu nos experimentos com a base do CTU-13. O que se nota, aliás, é que as características relativas ao protocolo TCP na verdade foram percebidas pelo método como dependentes de outras *features*. TcpAck e TcpFlagsAck, por exemplo, são dependentes de DestPort, provavelmente por não apresentarem variação de porta de destino nos pacotes TCP que continham informações sobre a *flag* ACK.

Outra possibilidade seria que nesse experimento essas *features* têm poder discriminante muito similar, o que permite a exclusão das que se referem ao protocolo TCP. Contudo, como o cenário proposto era sintético, com apenas uma máquina atuando como alvo e com um pequeno grupo de atacantes agindo, a variação de valores nas *features* não é tão expressiva. Isso pode ocasionar a inferência incorreta de algumas dependências, devido às poucas categorias (ou sejam, valores distintos) existentes para cada *feature*. Neste caso, encontra-se uma falha em nosso método, já que nele ainda não está presente um mecanismo que consiga distinguir essas duas situações apresentadas.

5. Conclusão

Os ataques DDoS volumétricos, por apresentarem ameaça a empresas e organizações, requerem métodos que permitam a sua detecção o mais rápido possível. Como é necessário

que a análise seja em tempo real, sob dados gerados de maneira contínua, as técnicas que somente trabalham com dados históricos não são as mais adequadas. Além disso, a partir do estudo dos trabalhos presentes na literatura, percebe-se a dificuldade existente em tratar de dados de ataques DDoS em algoritmos de seleção de *feature*, tendo em vista que muitas vezes não é possível ou viável utilizar as ferramentas mais tradicionais. Por isso, o método proposto neste trabalho visa oferecer um meio de realizar a análise de maneira *online*, sob dados em *streaming*.

Uma das maiores preocupações em lidar com esse tipo de dados é que a técnica precisa se adaptar à natureza dos mesmos, que são em sua maioria não-paramétricos e, frequentemente, esparsos. O nosso método mostra-se como uma opção interessante para análise rápida e em tempo real em casos que o volume do tráfego do ataque DDoS é bastante alto e o conteúdo das *features* variam dentro da janela de análise. Uma das principais vantagens do método é que é utilizado um critério simples e eficaz, que é a independência de variáveis, para selecionar as *features* mais representativas. Ademais, destaca-se a capacidade de selecionar *features* discriminativas especialmente nos conjuntos cujos dados possuem grande variação de valores, que é o caso da maioria das situações no mundo real.

Para os trabalhos futuros, procura-se explorar a maior falha do método percebida até então que impede a apresentação de resultados totalmente satisfatórios em bases de caráter sintético, com poucos atacantes e/ou poucos alvos do ataque. E há intenção de verificar se o vetor de características retornado pelo método é capaz de reduzir o erro de classificação, nas técnicas baseadas em aprendizagem de máquina.

Referências

[Bellman 1966] Bellman, R. (1966). Dynamic Programming. *Science*, 153(3731):34–37.

[Borini 2018] Borini, G. (2018). Primeiro trimestre tem 1,4 milhão de ataques DDoS, aponta levantamento. https://computerworld.com.br/2018/05/08/primeiro-trimestre-tem-14-milhao-de-ataques-ddos-aponta-levantamento/.

[Eskandari and Javidi 2016] Eskandari, S. and Javidi, M. M. (2016). Online streaming feature selection using rough sets. *International Journal of Approximate Reasoning*, 69:35–57.

[Freedman et al. 1998] Freedman, D., Pisani, R., and Purves, R. (1998). *Statistics*. W.W. Norton.

[García et al. 2014] García, S., Grill, M., Stiborek, J., and Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123.

[Guyon and Elisseeff 2003] Guyon, I. and Elisseeff, A. (2003). An Introduction to Variable and Feature Selection 1 Introduction. *An Introduction to Variable and Feature Selection*, 3(Mar):1157–1182.

[Guyon and Elisseeff 2008] Guyon, I. and Elisseeff, A. (2008). An Introduction to Feature Extraction. In *Feature Extraction*, pages 1–25.

[Hu et al. 2018] Hu, X., Zhou, P., Li, P., Wang, J., and Wu, X. (2018). A survey on online feature selection with streaming features.

[Idhammad et al. 2018] Idhammad, M., Afdel, K., and Belouch, M. (2018). Semi-supervised machine learning approach for DDoS detection. *Applied Intelligence*, 48(10):3193–3208.

[Jialei Wang et al. 2014] Jialei Wang, Peilin Zhao, Hoi, S. C. H., and Rong Jin (2014). Online Feature Selection and Its Applications. *IEEE Transactions on Knowledge and Data Engineering*, 26(3):698–710.

[Kim et al. 2006] Kim, H., Robert, C. P., and Casella, G. (2006). *Monte Carlo Statistical Methods*. Number 4.

[Li et al. 2017] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2017). Feature selection: A data perspective. *ACM Comput. Surv.*, 50(6):94:1–94:45.

[Osanaiye et al. 2016] Osanaiye, O., Cai, H., Choo, K. K. R., Dehghantanha, A., Xu, Z., and Dlodlo, M. (2016). Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. *Eurasip Journal on Wireless Communications and Networking*, 2016(1):130.

[Perkins et al. 2003] Perkins, S., Lacker, K., and Theiler, J. (2003). 10.1162/153244303322753698. *The Journal of Machine Learning Research*, 1(Mar):1333–1356.

[Rizzo 2007] Rizzo, M. (2007). *Statistical Computing with R*. Chapman & Hall/CRC The R Series. Taylor & Francis.

[Santos et al. 2017] Santos, A. A., Nogueira, M., and Moura, J. M. F. (2017). A stochastic adaptive model to explore mobile botnet dynamics. *IEEE Communications Letters*, 21(4):753–756.

[Sharafaldin et al. 2018] Sharafaldin, I., Habibi Lashkari, A., and Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. pages 108–116.

[Wu et al. 2013] Wu, X., Yu, K., Ding, W., Wang, H., and Zhu, X. (2013). Online feature selection with streaming features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(5):1178–1192.

[Yamada and Goto 2012] Yamada, R. and Goto, S. (2012). Using abnormal TTL values to detect malicious IP packets. *Proceedings of the Asia-Pacific Advanced Network*, 34(0):27.

[Zhou et al. 2005] Zhou, J., Foster, D., Stine, R., and Ungar, L. (2005). Streaming feature selection using alpha-investing. In *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining - KDD '05*, page 384, New York, New York, USA. ACM Press.

XXIV Workshop de Gerência e Operação de Redes e Serviços
SBRC 2019
Sessão Técnica 2

Uma Abordagem Não Supervisionada para Inferir Qualidade de Experiência em Redes Sem Fio de Grande Escala

Diogo Menezes Ferrazani Mattos, Dianne Scherly Varela de Medeiros, Natalia Castro Fernandes e Luiz Claudio Schara Magalhães

[1]MídiaCom - Departamento de Engenharia de Telecomunicações
Universidade Federal Fluminense (UFF)

Resumo. *Inferir a qualidade da experiência de usuários de redes sem fio é desafiador, pois o monitoramento da rede não captura a qualidade para cada usuário individualmente. Este artigo propõe uma abordagem não supervisionada, baseada em aprendizado de máquina, para inferir a qualidade de experiência de diferentes perfis de uso de uma rede sem fio de grande escala. A abordagem proposta usa a correlação entre dados de uso de pontos de acesso e estatísticas de fluxos de dados na rede. A ideia central da proposta é coletar dados de utilização de diversos pontos de acesso, correlacioná-los com as estatísticas dos fluxos das conexões que passam pelos pontos de acesso, reportados pelo NetFlow, e, a partir da aplicação do algoritmo de agrupamento* k-means*, inferir diferentes perfis de uso da rede. A abordagem proposta foi avaliada na rede sem fio real de grande escala e os resultados mostram que a separação dos fluxos em cinco agrupamentos permite identificar perfis característicos de estados degradados da rede e situações de sobrecarga em pontos de acesso, considerando apenas as estatísticas de fluxos reportadas.*

Abstract. *Inferring the quality of experience of wireless users is challenging, as network monitoring does not capture quality perception for each user individually. In this paper, we propose an unsupervised approach, based on machine learning, to infer the quality of experience from different usage profiles of a large-scale wireless network. The proposed approach uses the correlation between access point usage data and summaries of data flows in the network. The main idea of the proposal is to collect data on the usage of several access points, to correlate them with summaries of the connections flows that pass through the access points, reported by NetFlow, and to infer, from the application of the k-means clustering algorithm, different network usage profiles. The proposed approach was evaluated in the real wireless network, and the results show that the division of the flows in five clusters allows identifying characteristic profiles of the degraded state of the network and overload conditions in access points, considering only the flow summaries.*

1. Introdução

O uso de dados em dispositivos móveis vem crescendo a cada ano. Em 2016, as redes móveis contaram com um crescimento de 63%, alcançando um crescimento acumulado de dezoito vezes nos últimos 5 anos. Esse crescimento é marcado pela terceirização

Este trabalho foi realizado com recursos do CNPq, CAPES, RNP e FAPERJ.

do tráfego de dados de dispositivos móveis para redes fixas sem fio, que compõe 60% do tráfego móvel global [Cisco, 2017]. Assim, da popularização das redes sem fio padrão IEEE 802.11 as tornam presença constante em diversos ambientes [Mattos et al., 2019, Divgi e Chlebus, 2013]. No entanto, essa mesma popularização fomenta o desafio de controlar e gerenciar esse tipo de rede, já que um ponto de acesso típico está na vizinhança de diversos outros, sofrendo e gerando interferência [Biswas et al., 2015]. A ausência de mecanismos de inteligência e de ação rápida na rede, muitas vezes associada a visões limitadas de ferramentas de controle por fluxo, prejudicam a qualidade de experiência (QoE) de usuários de redes sem fio de grande escala [Jang et al., 2017].

A QoE do usuário de uma rede sem fio é difícil de mensurar, pois o serviço de rede percebido pelo usuário está sujeito a diversas interferências e ao tipo de conteúdo acessado. Assim, a inferência da QoE implica a necessidade de identificação de perfis de uso da rede sem fio. Em especial, a identificação do perfil de uso da rede relaciona-se com a maneira como o ponto de acesso e os usuários conectados a ele interagem com o restante da rede. Ademais, a inferência de perfis do uso de dados em redes sem fio permite definir padrões de uso [Zhang et al., 2016, Ghosh et al., 2011] e propicia que a cobrança pelo uso da rede seja de acordo com a classificação do uso [Sen et al., 2013]. Dessa forma, inferir padrões de uso da rede sem fio que considerem tanto os dados trafegados como informações de associação de usuários a pontos de acesso é essencial para a identificação de pontos de falhas e gargalos de desempenho em redes sem fio de grande escala.

Este artigo propõe uma abordagem de inferência de qualidade de experiência de usuários de rede sem fio de grande escala usando uma abordagem não supervisionada. A ideia central da proposta é gerar um conjunto de dados, correlacionando estatísticas de fluxos de cada usuário da rede sem fio com características dos pontos de acesso a que estão associados e, a partir do conjunto de dados correlacionados, extrair agrupamentos de usuários com perfis semelhante para finalmente identificar a qualidade de experiência de cada perfil extraído da rede. Para tanto, a abordagem proposta conta com uma coleta de dados de estatísticas de fluxos NetFlow, um módulo de coleta de dados da interface sem fio dos pontos de acesso da rede, um módulo de correlação de dados e, por fim, um módulo de processamento e inferência dos perfis. A inferência de perfis de uso é calculada através do algoritmo de agrupamento de dados não supervisionado *k-means*.

Trabalhos anteriores focam em modelar a chegada de novos usuários em redes sem fio [Ghosh et al., 2011], identificar possíveis gargalos de desempenho na interface sem fio da rede [Biswas et al., 2015], caracterizar o uso de dados por aplicações móveis [Qian et al., 2011, Shye et al., 2010] e analisar o desempenho em cenários restritos [Oliveira et al., 2016, Magalhães e Mattos, 2018]. Diferentemente, a proposta deste artigo é coletar tanto dados de tráfego com NetFlow, quanto métricas de associação de usuários em pontos de acesso, correlacionar os dados e, então, inferir perfis de uso. O conjunto de dados[1] gerados se baseia na rede sem fio institucional da Universidade Federal Fluminense que conta com 547 nós, atende a mais de 60 mil usuários e conta com picos de mais de 5.000 usuários conectados simultaneamente. A avaliação da proposta mostra que foram identificados cinco perfis de uso distintos da rede e as características de cada perfil relacionam-se com o tipo de serviço acessado, bytes trafegados, duração dos fluxos e a carga de uso sobre os pontos de acesso.

[1]Os dados coletados são referentes ao período de 17 a 24 de abril de 2018.

O restante do artigo está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta o problema de caracterização de perfis de uso da rede. A abordagem de inferência de qualidade de experiência é exposta na Seção 4. A proposta é avaliada no cenário de uma rede sem fio de grande escala real e os resultados são discutidos na Seção 5. A Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

As aplicações de dispositivos móveis, como telefones inteligentes (*smartphones*), são as principais geradoras de fluxos em redes sem fio, em especial devido à terceirização dos tráfegos das redes de celular para a rede sem fio infraestruturada [Joe-Wong et al., 2013]. Manweiler *et al.* propõem um sistema de predição do tempo de estadia de clientes em pontos de conexão sem fio. A ideia central do sistema é aprender assinaturas de um conjunto de clientes inicial no ponto de acesso e, então, inferir o tempo de permanência de cada cliente nas proximidades do ponto de acesso sem fio. A predição utiliza diversos sensores nos celulares dos clientes, gerando uma matriz de dados que é passada para um classificador de máquina vetor de suporte (*Support Vector Machine* - SVM) que separa os clientes em classes de comportamento predeterminadas. As predições são geradas em série [Manweiler et al., 2013].

Quian *et al.* desenvolvem a ferramenta ARO (*Application Resource Optimizer*) que visa inferir e otimizar o uso de recursos por aplicações na interface de rádio de dispositivos móveis [Qian et al., 2011]. A proposta se baseia em uma abordagem entre camadas, considerando tanto a aplicação quanto o controle na camada de rádio. Em especial, a proposta desenvolve uma técnica para estimação dos estados de controle da interface de rádio e presume causas para gargalos no uso de tráfego de dados. A proposta extrai perfis de uso de aplicações populares e permite identificar que algumas aplicações têm interações ineficientes com a camada de rádio e, por conseguinte, implica gargalos no tráfego de dados e redução de desempenho nos dispositivos móveis. Por sua vez, Quiang *et al.* identificam perfis de comportamento de uso de aplicações em dispositivos móveis [Xu et al., 2011], mas, diferentemente do trabalho anterior, os autores focam no padrão de uso dos dispositivos e não no consumo de recursos de cada aplicação. Esses trabalhos relacionam-se com a inferência de perfis de uso em redes sem fio na medida em que os dispositivos móveis são os mais numerosos na rede e as aplicações mais prevalentes geram assinaturas de consumo de dados que impactam a rede sem fio de formas distintas [Shye et al., 2010].

Oliveira *et al.* argumentam que entender as atividades dos usuários em redes de acesso sem fio é essencial para garantir a escalabilidade da rede e para atender às futuras demandas de tráfego [Oliveira et al., 2016]. Portanto, os autores investigam e caracterizam as atividades de usuários em redes sem fio e comparam os padrões encontrados entre redes universitárias e redes em áreas urbanas. Os resultados das análises evidenciam que há uma relação linear entre o número de sessões de acesso sem fio e o número de pontos de acesso, mas a relação não é mantida ao se considerar em quantos pontos de acesso um usuário se conecta. Paralelamente, Biswas *et al.* verificam que a monitoração de redes sem fio exige monitorar o espectro usado por cada rede para inferir a interferência gerada por cada uma delas, pois o uso dos canais de redes WiFi é baixo perante o número de redes presentes [Biswas et al., 2015]. Tais resultados são ratificados ao analisar os dados da Universidade Federal Fluminense [Magalhães e Mattos, 2018]. Magalhães e Mattos

evidenciam que muitos usuários tendem a se associar a poucos pontos de acesso durante um dia e que, embora o número de redes sem fio no entorno de um ponto de acesso seja substancialmente alto, a interferência que cada rede gera depende da carga a que está submetida. De forma semelhante, Ghosh *et al.* caracterizam o acesso sem fio em redes de grande escala públicas e propõem um modelo de chegada de usuários [Ghosh et al., 2011]. Os autores combinam um agrupamento estático com regressão de Poisson para modelar o processo de Poisson não-estacionário que define a chegada de novos clientes sem fio. A ideia central é entender a carga típica de redes sem fio de grande escala.

Divgi *et al.* revisam a literatura sobre a caracterização de redes sem fio e analisam os dados de acesso de uma rede sem fio de âmbito nacional gerados em um período de 5 meses [Divgi e Chlebus, 2013]. Os autores caracterizam o uso comercial de redes sem fio, categorizando os usuários pelo tipo de conta de acesso, desenvolvendo métricas que mostram a flutuação da população de usuários e propondo novas métricas para contabilizar tempo de sessão e uso de dados. Já Zhang *et al.* buscam por padrões de sequência de uso em redes sem fio [Zhang et al., 2016]. Os autores inferem a sequência de padrões através de um modelo oculto de Markov. Para aplicar o modelo, os dados são divididos em agrupamentos de usuários usando o método de *k-metoids*. O número ideal de agrupamentos é definido usando a estatística de intervalo (*Gap Statistic*). Nenhum dos trabalhos mencionados relaciona estatísticas de fluxos e métricas de uso da interface sem fio.

Jang *et al.* propõem o sistema RFlow$^+$ [Jang et al., 2017]. Os autores argumentam que a ausência de mecanismos inteligentes e que possam tomar ações rápidas nas redes sem fio associada a ferramentas de monitoramento de tráfego que geram visões limitadas sobre a rede prejudicam a confiabilidade das redes sem fio. Assim, o sistema RFlow$^+$ combina a ideia de redes definidas por *software* e monitoramento de redes sem fio para prover um arcabouço de gerenciamento e controle que permita a tomada de ações rápidas na rede. Para tanto, os autores propõem dois níveis de contadores de dados de fluxo, um local e outro global. Os dados desses contadores de fluxos são usados no controle da rede. Paralelamente, Dely *et al.* argumentam que devido à sobreposição da área de cobertura de pontos de acesso de uma mesma rede, a otimização da rede pode causar a troca constante de associação entre usuário e ponto de acesso [Dely et al., 2012]. Assim, os autores introduzem um problema de otimização linear inteira mista que visa reduzir o números de trocas de reassociação ao considerar o custo da reassociação. Balbi *et al.* abordam os problemas da instabilidade na associação propondo um processo simples de suavização da série temporal de amostras de sinal de recepção [Balbi et al., 2016]. Diferentemente, a proposta deste artigo aborda o gerenciamento de redes sem fio sob a perspectiva da análise de dados e correlaciona os dados de fluxos unitários, colhidos por NetFlow, a dados de associação de usuários a pontos de acesso. A proposta analisa o conjunto de dados contendo tanto dados da interface sem fio quanto os fluxos na rede de transporte para extrair conhecimento útil no controle e monitoramento das redes sem fio.

3. A Inferência de Perfis de Uso da Rede

Inferir perfis de uso da rede consiste em analisar os dados de uso da rede e descobrir padrões que se repetem em diferentes usuários. Os perfis podem apontar usos característicos de determinadas aplicações [Shye et al., 2010] e o quanto de recursos é consumido por cada aplicação [Qian et al., 2011]. Contudo, em trabalhos anteriores a busca por padrões de uso de aplicações ou padrões de uso de recursos de rede são realizadas de

forma supervisionada, isto é, conjuntos de dados característicos de cada aplicação são usados para treinar um classificador capaz de reconhecer os padrões em um conjunto de dados de teste [Andreoni Lopez et al., 2017]. Ao identificar padrões anteriormente ocultos de uso da rede é possível extrair não somente conhecimento a respeito do funcionamento da rede [Biswas et al., 2015, Ghosh et al., 2011], mas também a respeito dos usuários, como preferências e padrões de mobilidade [Wang et al., 2014, Guo et al., 2014].

O problema da inferência de perfis não supervisionada é complexo, pois *a priori* não são conhecidos os padrões de uso da rede e, assim, são usados algoritmos de agrupamento não supervisionado para identificação de padrões recorrentes [Zhang et al., 2016]. O agrupamento de amostras de dados em conjuntos que definem um padrão é um problema NP-difícil [Zhang et al., 2016]. Os algoritmos de aprendizado de máquina para agrupamento não supervisionado visam buscar vetores que permitam representar agrupamentos de dados com maior espaçamento entre eles e, simultaneamente, reduzindo a distância das amostras até o centro dos agrupamentos. O algoritmo de agrupamento *k-means*, por exemplo, busca k vetores que minimizam a distância de todas as amostras aos vetores centrais de cada agrupamento. O algoritmo atribui o vetor de dados a exatamente um agrupamento. A cada rodada, o algoritmo recalcula o centro de cada agrupamento e termina quando as atribuições aos agrupamentos não mudam entre duas rodadas.

Ao se considerar uma rede sem fio de grande escala, como a rede institucional da Universidade Federal Fluminense, a detecção de padrões de uso torna-se computacionalmente custosa devido à grande massa de dados gerada pelo uso contínuo da rede. Outro fator complicador é que os dados necessários para inferir a qualidade de experiência dos usuários são dispersos em diversas fontes de registros de atividades (*logs*) não correlacionadas. Vale ressaltar que na rede considerada há 547 pontos de acesso, gerando registros de associação e desassociação de usuários, e os fluxos de dados gerados pelos usuários são reportados pelos roteadores de saída da rede usando NetFlow. Contudo, as informações de associação de usuários não se relacionam diretamente com os relatórios NetFlow, já que as informações de associação carregam dados da camada de enlace, enquanto os relatórios NetFlow carregam dados das camadas de rede e superiores. Assim, a variedade de fontes de dados é um desafio para inferir os perfis de uso em redes de grande escala. Dessa forma, com dados sendo gerados em alta velocidade, grande massa de dados e variabilidade nos dados, a inferência de perfis de uso é um problema de *Big Data*.

4. A Abordagem Proposta de Geração de Perfis do Uso da Rede Sem Fio

A rede sem fio institucional da Universidade Federal Fluminense (UFF) é distribuída por mais de 90 prédios em 16 *campi* universitário distintos. A infraestrutura da rede sem fio é baseada no *software* livre SCIFI (Sistema de Controle Inteligente para redes sem fio), originado de um grupo de trabalho da RNP (Rede Nacional de Ensino e Pesquisa) - o GT SCIFI. O SCIFI é composto por um controlador de *software* e um *firmware* instalado em pontos de acesso de baixo custo. O *software* controlador configura os pontos de acesso para usarem canais que minimizem a interferência entre pontos de acesso SCIFI e, também, evitem a interferência com outras redes [Balbi et al., 2012, Magalhães e Mattos, 2018]. A infraestrutura de rede conta com 5 *gateways* de saída de tráfego da rede sem fio para a Internet. Os *gateways* de conexão da rede sem fio com a Internet concentram o tráfego dos maiores *campi*. Este trabalho foca na análise do tráfego de um *campus* da UFF, Praia Vermelha, no período de 17 a 24 de

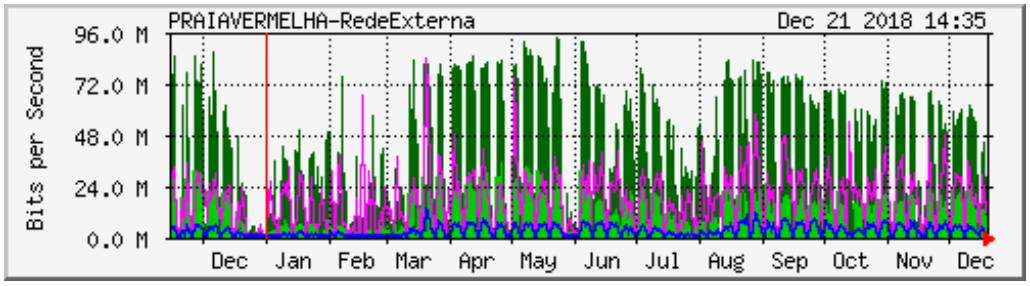


Figura 1. Tráfego encaminhado no *campus* Praia Vermelha da Universidade Federal Fluminense no ano de 2018.

abril de 2018, identificando um total de 6.770 dispositivos únicos que acessaram a rede através de um subconjunto de 363 pontos de acesso. A Figura 1 mostra o tráfego encaminhado no *campus* Praia Vermelha. A infraestrutura nesse *campus* realiza o roteamento de pacotes a taxas de pico próximas a 100 Mb/s. Vale ressaltar que os dados da Figura 1 representam somente o uso de dados da rede sem fio de um único *campus* [2]. No contexto da rede institucional da UFF, a abordagem proposta consiste em analisar os dados de fluxos gerados pelos usuários da rede e correlacionar com dados de associação entre usuários e ponto de acesso. Dessa forma, o *firmware* e o *software* de controle SCIFI são essenciais para a coleta de dados.

A abordagem proposta se baseia na análise de estatísticas de fluxos da rede. O fluxo de rede pode ser definido como uma sequência unidirecional ou bidirecional de pacotes entre dois nós da rede com algum atributo em comum. Comumente, define-se os atributos em comum como a 5-tupla: endereços de origem e destino, protocolo de transporte e portas de origem e destino [Andreoni Lopez et al., 2017, Mattos et al., 2019]. Os campos capturados na descrição dos fluxos da rede fornecem um conjunto avançado de estatísticas de tráfego, incluindo usuário, protocolo, porta e tipo de serviços que podem ser usados para uma análise ampla com diferentes propósitos, como segurança de rede, monitoramento de rede, análise de tráfego, planejamento de capacidade, classificação de tráfego, contabilidade e faturamento. Ao se considerar a caracterização através da tecnologia NetFlow, o processo geral inclui capturar, amostrar, gerar, exportar, coletar, analisar e visualizar [Li et al., 2013].

NetFlow é uma tecnologia de monitoramento de tráfego desenvolvida na Cisco e define como um roteador exporta informações e estatísticas de fluxos de rede. Os dispositivos de rede examinam os pacotes que chegam nas interfaces e capturam estatísticas de tráfego por fluxo com base na configuração para amostragem ou filtragem, criam um cache do fluxo, agregam e exportam os dados por meio dos protocolos UTP ou SCTP. A entrada de cache do NetFlow é criada pelo primeiro pacote de um fluxo, mantida para características de fluxo semelhantes e exportada para coletores periodicamente com base em temporizadores de fluxo ou no gerenciamento de cache de fluxo. Ao usar amostragem em NetFlow, são introduzidos alguns novos desafios, tais como não contabilizar novos fluxos quando o cache estiver cheio; a redução da precisão na amostragem dos fluxos, especialmente quando a taxa de amostragem é ajustada pela taxa de tráfego; e discrepância na ordem em que os registros de fluxo são exportados em relação à ordem em que o tráfego

[2]Os dados apresentados são reportados pela ferramenta MRTG que consolida informações colhidas por SMNP. Os dados estão disponíveis em http://200.20.0.200/mrtg.

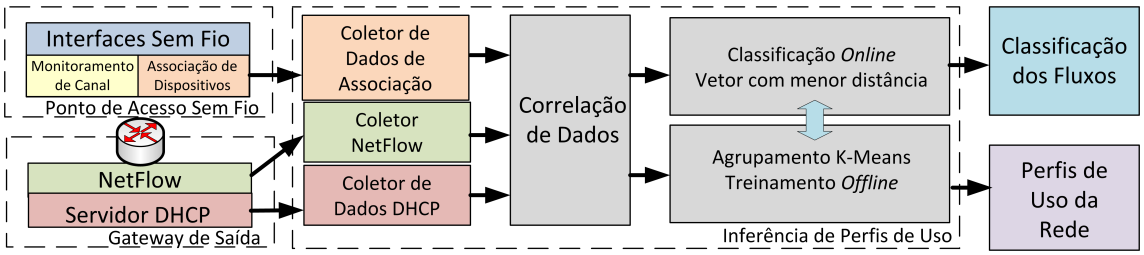


Figura 2. A abordagem proposta consiste na coleta e correlação de estatísticas de fluxos NetFlow e dados de associação de clientes móveis com pontos de acesso. Os dados são correlacionados para a geração de resumos bidirecionais de fluxos e a identificação do dispositivo móvel e do ponto de acesso que geram os dados. Os módulos de processamento *online* e *offline* geram inferências em tempo real do perfil de uso de cada fluxo na rede.

do fluxo chega ao roteador. Existem variedades de coletores NetFlow e ferramentas de análise de fornecedores comerciais. Neste artigo, todos os fluxos são contabilizados, pois é realizada uma amostragem individual de cada fluxo e a taxa de transmissão de linha é inferior à capacidade de encaminhamento do *gateway* de conexão com a Internet, já que o tráfego atinge 100 Mb/s, enquanto o *gateway* tem capacidade para rotear até 1 Gb/s.

A abordagem proposta é evidenciada na Figura 2 e consiste de módulos de coleta de dados, módulos de correlação e módulos de processamento *online* e *offline*. Após o processamento dos dados, o conhecimento extraído é analisado. Os módulos de **coleta de dados** são divididos em duas partes. A primeira parte executa nos elementos de rede, sejam os pontos de acesso, sejam os *gateways* de acesso à Internet, e é responsável por executar a coleta local de dados. Nos pontos de acesso, são coletados os dados de associação de novos dispositivos através de mensagens do *hostapd*[3], o ponto de acesso por *software* executado pelo *firmware* SCIFI. Os *gateways* realizam a coleta de dois conjuntos de dados importantes, os registros de entrega de endereços IP pelo DHCP e as estatísticas dos fluxos roteados na rede. A coleta dos dados locais gerados nos pontos de acesso e dos dados de entrega de IP nos *gateways* é realizada pelo *rsyslog* [4]. Já a coleta das estatísticas NetFlow é realizada pela ferramenta de coleta e análise `SiLK` [5]. Após a coleta, todos os dados são concentrados em um servidor de gerenciamento.

O módulo de **correlação de dados** limpa, funde e correlaciona os dados gerados pelas diversas fontes. O foco desse módulo é gerar, para cada registro de fluxo reportado pelo NetFlow, características extras, como a identificação do dispositivo do cliente que realiza o fluxo, o ponto de acesso em que o cliente está conectado no momento em que realiza o fluxo e o número de clientes ativos, ou seja, que estão com fluxos ativos, no mesmo ponto de acesso. Para considerar o número de clientes ativos, foi avaliado o número de fluxos ativos de clientes distintos em uma janela de 10 s. Ao realizar a fusão de dados, o módulo agrega mais informações à análise e permite identificar correlações entre características distintas.

O módulo de **processamento dos dados** é particionado em processamento *online* e em processamento *offline*. O processamento *online* realiza classificação dos registros en-

[3]Disponível em http://w1.fi/hostapd/.
[4]Disponível em https://www.rsyslog.com/.
[5]Disponível em https://tools.netsa.cert.org/silk/.

trantes conforme chegam ao módulo. Essa classificação é realizada de acordo com os vetores de identificação dos agrupamentos de dados calculados pelo processamento *offline*. O processamento *offline* realiza a análise de dados históricos armazenados. Para tanto, nesse trabalho, a análise realizada é não supervisionada e consiste no pré-processamento dos dados e na aplicação do algoritmo *k-means* para realizar o agrupamento. O pré-processamento realizado exclui características textuais, constantes ou vazias reportadas tanto pelo NetFlow quanto pelo módulo de correlação de dados.

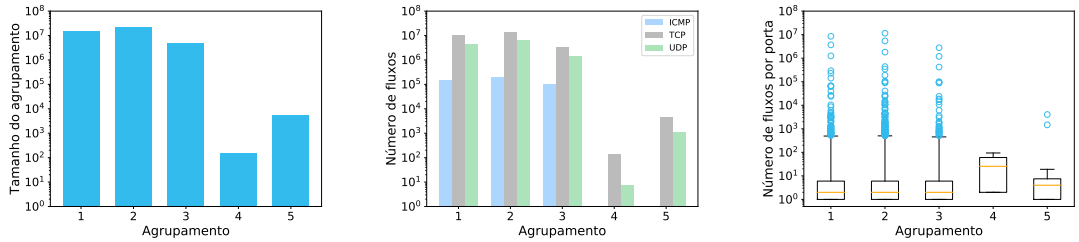
Após o processamento dos dados, é possível identificar diferentes padrões de uso mais prevalentes na rede. Assim, a proposta reporta a classificação dos fluxos *online*, *i.e.*, assim que são correlacionados e classificados. Ademais, quando executada em tempo diferenciado sobre uma base histórica de uso da rede, a abordagem proposta permite identificar padrões prevalentes na rede que são as inferências de perfis de uso da rede. A inferência desses perfis implica em maior distância[6] entre os vetores que representam cada agrupamento. Sendo assim, os perfis inferidos são os que permitem maior diferenciação do uso da rede. Vale ressaltar que a inferência de qualidade de experiência está associada à análise dos perfis representados pelos agrupamentos identificados pelo algoritmo *k-means*. O algoritmo *k-means* é usado pois é simples e não paramétrico, o que permite a extração mais imediata de perfis sem o conhecimento prévio dos dados. A qualidade da experiência consiste em analisar os dados de cada agrupamento e inferir o que os clientes classificados naquele perfil têm em comum. Em especial, verifica-se a quantidade de clientes associados a um mesmo ponto de acesso como métrica de competição por recursos e a banda de transmissão média alcançada pelos clientes em cada agrupamento.

5. Resultados Experimentais

A avaliação da abordagem proposta foi realizada sobre os dados coletados da rede sem fio do *campus* Praia Vermelha, da Universidade Federal Fluminense. Para tanto, os pontos de acesso foram configurados para reportar os dados para o servidor responsável por executar o *software* controlador SCIFI da rede. Os dados coletados foram consolidados e processados em um servidor equipado com processador Intel Core i5 4460, com 4 núcleos de processamento, e 32 GB de memória RAM. O processamento dos dados *offline* foi implementado em Python com uso das bibliotecas Pandas[7] e Sklearn[8] e com apoio da ferramenta KNIME 3.5.3[9]. Em uma análise preliminar, foi executada a estatística de intervalo (*Gap Statistic*) para determinar o número de agrupamentos ideal [Zhang et al., 2016] para descrever o conjunto de dados. A estatística de intervalo busca o número de agrupamentos que maximiza a distância entre os agrupamentos, enquanto minimiza a distância entre as amostras em um mesmo agrupamento. A estatística é projetada para ser agnóstica ao algoritmo de agrupamento utilizado. A estatística de intervalo foi executada em uma amostra do conjunto de dados e resultou em 5 agrupamentos. Portanto, nas demais avaliações foram considerados 5 agrupamentos.

As análises a seguir são apresentadas prioritariamente em relação aos fluxos de descarga de conteúdos da Internet para a rede sem fio. A análise foca nos fluxos de descarga, pois como os usuários da rede sem fio estão atrás de *firewall* e usando endereços não

[6]A proposta considera a distância euclidiana como métrica para indicar a separação entre agrupamentos.
[7]Disponível em https://pandas.pydata.org/.
[8]Disponível em https://scikit-learn.org/stable/.
[9]Disponível em https://www.knime.com/.



(a) Tamanho em número de fluxos dos agrupamentos.

(b) Distribuição de protocolos por agrupamento.

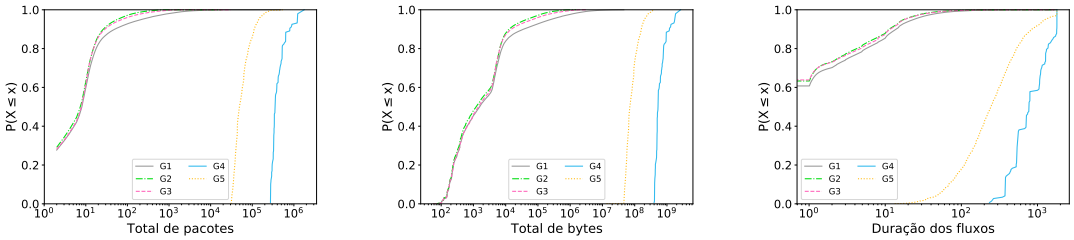
(c) Distribuição do uso de portas por agrupamento.

Figura 3. Agrupamentos calculados pelo algoritmo *k-means*. Gráficos apresentados em escala logarítmica. a) Os agrupamentos 1, 2 e 3 são os mais prevalentes, concentrando maior número de fluxos. b) Os perfis de tráfego dos agrupamentos 1, 2 e 3 são semelhantes, concentrando fluxos TCP, UDP e ICMP. c) Distribuição do números de fluxos em cada porta nos agrupamentos calculados. Os *outliers* indicam que poucas portas têm número de fluxos muito superior à média, indicando a tendência no comportamento do agrupamento.

roteáveis locais fornecidos por um NAT (*Network Address Translation*), não é possível que um nó externo à rede sem fio abra uma conexão com um usuário da rede sem fio. Ademais, caso existam fluxos entre usuários da rede sem fio, esses fluxos não são reportados pelo NetFlow, já que a coleta de dados é realizada pelo *gateway* de saída da rede sem fio para a Internet.

Após a execução do algoritmo *k-means*, os dados do conjunto de dados foram classificados de forma não supervisionada nos 5 agrupamentos. Cada agrupamento define um perfil de uso da rede. Três agrupamentos contêm da ordem de 10^7 fluxos em cada, conforme mostrado na Figura 3(a), enquanto os dois outros agrupamentos contêm menos fluxos, da ordem de 10^4 e 10^2 fluxos. Quando comparadas as proporções do uso de diferentes protocolos de transporte em cada agrupamento, conforme Figura 3(b), verifica-se que o TCP é o protocolo mais comum em todos os agrupamentos. Contudo, há uma presença constante de sinalização ICMP nos agrupamentos 1, 2 e 3. Nos três agrupamentos, o número de fluxos ICMP é semelhante. Diferentemente, nos agrupamentos 4 e 5 os fluxos do protocolo ICMP são ausentes e destaca-se que no agrupamento 4 há uma maior predominância de fluxos TCP, o que indica um perfil diferente do agrupamento 4 em relação aos demais. Na Figura 3(c), analisa-se o uso das portas de destino dos fluxos na rede, pois as portas de destino evidenciam quais serviços são acessados pelos usuários da rede sem fio. Verificou-se que as portas mais prevalentes em todos os fluxos são as UDP 53, TCP 80 e TCP 443 [Andreoni Lopez et al., 2017]. Tal comportamento é o esperado, pois os acesso mais comuns são a conteúdo *web* ou a serviços que usam chamadas *web*, tais como aplicativos de redes sociais ou mensagens instantâneas. O comportamento de maior uso dessas portas nos agrupamentos é marcado pela presença de *outliers* na Figura 3(c). De forma semelhante à análise anterior, os agrupamentos 4 e 5 se destacam por apresentarem menos *outliers* e uma distribuição maior do uso de portas.

Ao avaliar a distribuição de probabilidades de número de pacotes por agrupamento, Figura 4(a), *bytes* trafegados por cada fluxo no agrupamento, Figura 4(b), e a duração dos fluxos, Figura 4(c), verifica-se uma clara diferenciação entre os fluxos nos agrupamentos 1, 2 e 3 para os demais fluxos nos agrupamentos 4 e 5. Os fluxos nos agru-

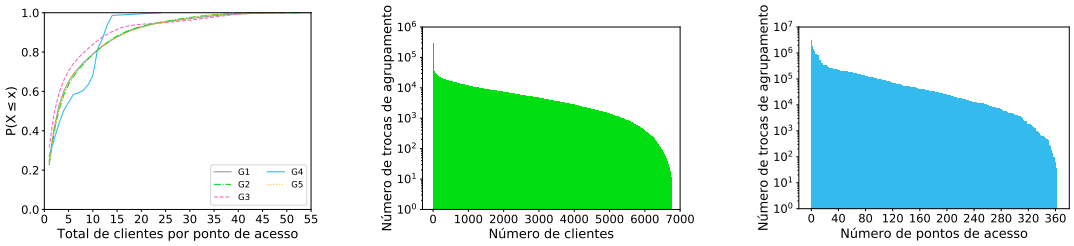


(a) Distribuição do número de pacotes por agrupamento.

(b) Distribuição do tamanho em bytes por agrupamento.

(c) Distribuição da duração dos fluxos por agrupamento.

Figura 4. Probabilidade cumulativa da distribuição de caracterização dos agrupamentos. a) Agrupamentos 1, 2 e 3 tendem a ter menos pacotes que os agrupamentos 4 e 5. b) De forma semelhante, os agrupamentos 4 e 5 são responsáveis pelo tráfego de maior quantidade de *bytes*. c) O agrupamento 4 é o que tende a ter os fluxos mais duradouros.



(a) Distribuição do número de clientes por ponto de acesso em cada agrupamento.

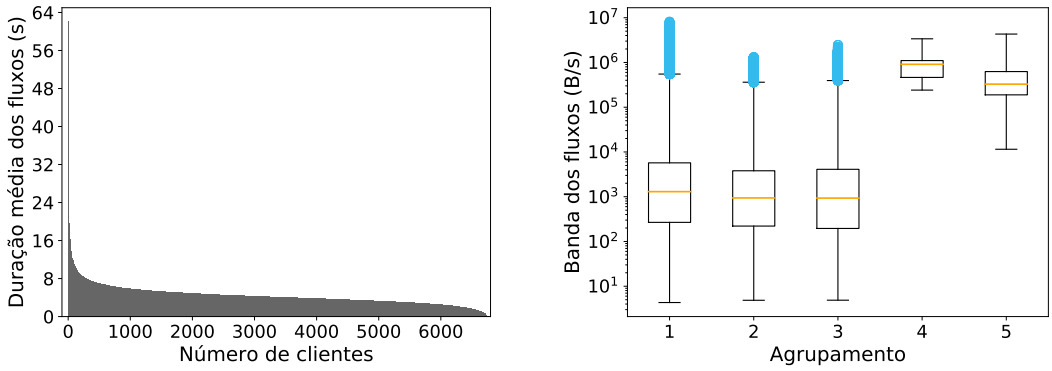
(b) Número de trocas de classificação para cada cliente.

(c) Número de trocas de classificação para cada ponto de acesso.

Figura 5. Caracterização do comportamento dos clientes nos agrupamentos. a) Probabilidade cumulativa do número de clientes conectados a um mesmo ponto de acesso por agrupamento. b) Número de vezes que um cliente tem seu perfil recalculado como outro agrupamento na rede. c) Número de vezes que um ponto de acesso tem o agrupamento de que participa trocado.

pamentos 1, 2 e 3 tendem a ser mais curtos e, portanto, trafegam menos pacotes e menor quantidade de *bytes*. Enquanto os fluxos dos agrupamentos 4 e 5 são fluxos mais duradouros com maior quantidade de *bytes* trafegados. Observando a diferença de duração entre os fluxos dos agrupamentos 4 e 5, verifica-se que a probabilidade de apresentar fluxos longos converge, ao passo que ao avaliar o número de pacotes e *bytes*, verifica-se que os fluxos do agrupamento 4 tendem a apresentar maior volume de dados. Essas características indicam que fluxos do agrupamento 4 tendem a apresentar maior taxa de transmissão que os do agrupamento 5.

Ao considerar o número de clientes que acessam um ponto de acesso simultaneamente, há uma diferenciação no comportamento dos agrupamentos calculados. A Figura 5(a) mostra que o agrupamento 3 tende a ter menos clientes conectados por ponto de acesso que os demais agrupamentos. O agrupamento 4, que conta com fluxos com maior duração do que os demais agrupamentos, tende a ter mais clientes conectados por ponto de acesso do que qualquer outro agrupamento em até 80% dos fluxos. Contudo, o agrupamento 4 atinge o limite de clientes conectados por ponto de acesso, com aproximadamente 25 clientes por ponto de acesso, acima desse valor, não há mais fluxos classificados no



(a) Duração média dos fluxos por cliente.

(b) Taxa de transmissão alcançada em cada agrupamento.

Figura 6. Avaliação dos perfis de uso. a) Distribuição da duração dos fluxos por clientes identificados na rede. A duração segue uma distribuição de cauda longa. b) Avaliação da taxa de transmissão alcançada em cada agrupamento indica que os agrupamentos 4 e 5 apresentam as taxas de transmissão mais elevadas.

agrupamento 4. Acima de 25 usuários o canal sem fio passa a ser um entrave para manter um fluxo ativo por grande período de tempo e com alta taxa de encaminhamento de dados. Esse resultado reforça a ideia de que a previsão máxima ideal de carga para pontos de acessos é entre 20 e 30 usuários. Os agrupamentos 1, 2 e 5 obtiveram resultados semelhantes quanto ao uso concomitante dos pontos de acesso.

As Figuras 5(b) e 5(c) evidenciam a troca de classificação entre agrupamentos por cliente e por ponto de acesso respectivamente. Vale ressaltar que o comportamento do número de trocas de classificação entre agrupamentos segue a mesma distribuição para clientes e para pontos de acesso. A métrica de trocas de classificação considera que a classificação por agrupamento é atribuída ao cliente ou ao ponto de acesso e, então, verifica ao longo do conjunto de dados quantas vezes há mudança na classificação atribuída ao cliente ou ao ponto de acesso. A ideia central dessa métrica é avaliar se há dependência entre a classificação e cliente ou ponto de acesso. No entanto, como a distribuição das trocas de classificação é semelhante entre clientes e pontos de acesso, verifica-se que a classificação é independente desses fatores, e, assim, a classificação é fortemente dependente dos fluxos que são realizados na rede.

A Figura 6(a) revela a duração média dos fluxos por cliente. Esse resultado evidencia que a duração dos fluxos segue uma distribuição de cauda longa, em que poucos clientes executam fluxos longos, enquanto a maioria dos clientes executam fluxos de curta duração. Tal resultado relaciona-se com a caracterização do tráfego da rede que aponta que os fluxos mais prevalentes são os de consulta DNS e fluxos de requisição HTTP e HTTPS. Por fim, foi avaliada a taxa de transmissão alcançada pelos fluxos em cada agrupamento. Os resultados mostrados na Figura 6(b) revelam que os agrupamentos 4 e 5 são os que apresentam as taxas de transmissão mais altas, com destaque para o agrupamento 4 que agrega fluxos com taxas de transmissão da ordem de 8 Mb/s. Os agrupamentos 1, 2 e 3 apresentam taxas de transmissão variáveis, com mediana próxima a 10 kb/s.

Vale notar que após a caracterização dos agrupamentos calculados pelo algoritmo *k-means* é possível definir que cada agrupamento representa um perfil de uso da rede sem

fio e, a partir das análises, define-se níveis de qualidade de experiência para os clientes. Assim, verifica-se que clientes que têm fluxos classificados nos agrupamentos 4 e 5 são aqueles que experimentam uma boa qualidade de serviço na rede. Em especial, os clientes que têm fluxos classificados nos agrupamentos 4 e 5 têm acesso a uma taxa de transmissão da ordem de alguns Mb/s. Ao verificar que um cliente tem fluxos classificados no agrupamento 4 é possível inferir, com mais de 90% de certeza, que o cliente está conectado em um ponto de acesso com menos de 25 clientes, dado o resultado exposto na Figura 5(a). Um cliente, ao ter seus fluxos classificados nos agrupamentos 1, 2 e 3, pode estar passando por dificuldades de conexão devido a excesso de uso na rede sem fio ou, simplesmente, pode estar usando serviços que exigem poucos recursos da rede sem fio, tais como redes sociais e aplicativos de mensagens. Esses serviços se caracterizam por fluxos de curta duração e pouco tráfego de dados. Portanto, o impacto na rede é reduzido. Esse comportamento é mais marcante em fluxos dos agrupamentos 2 e 3 que tendem a ser menores que os do agrupamento 1.

Outro ponto interessante ressaltado pela Figura 5(a) é que os fluxos do agrupamento 3 tendem a ser de clientes que estão associados a pontos de acesso com menos clientes conectados e, portanto, experimentam condições de rede melhores que os clientes dos agrupamentos 1 e 2. Sendo assim, é possível inferir que clientes que são classificados no agrupamento 4 são aqueles que experimentam as melhores condições da rede sem fio, seguidos pelos clientes do agrupamento 5 e, em seguida, pelos do agrupamento 3. Os clientes dos agrupamentos 1, 2 e 3 têm perfis de uso semelhantes, mas os clientes do agrupamento 3 experimentam redes menos congestionadas na interface sem fio, pois em 90% das vezes estão conectados em pontos de acesso abaixo da situação de sobrecarga. Os clientes dos agrupamentos 1 e 2 experimentam a rede em condições mais adversas que os demais, mas os clientes do agrupamento 2 atingem quantidades trafegadas de dados maiores que os do agrupamento 1, indicando uma melhor experiência na rede. Já os clientes do agrupamento 1 experimentam o uso da rede sem fio prioritariamente para uso de fluxos curtos e que trafegam pouca quantidade de dados.

6. Conclusão

O cálculo de perfis de uso da rede para inferir a qualidade de experiência de clientes somente com base em estatísticas de fluxos e dados de associação de dispositivos a pontos de acesso é um problema complexo. As estatísticas de fluxos não revelam questões importantes da rede sem fio como o uso do espectro e a quantidade de clientes que compartilham o mesmo canal. Esses desafios são ainda mais críticos em redes sem fio de grande escala em que há concentração de usuários em locais e períodos específicos. Este artigo propôs uma abordagem não supervisionada de geração de perfis de uso da rede sem fio para a inferência da qualidade de experiência do usuário final. Para tanto, a abordagem proposta correlaciona dados de diversas fontes, como estatísticas de fluxos NetFlow e dados de associação de clientes a pontos de acesso reportados pelos próprios pontos de acesso. A correlação dos dados ainda usa relatórios da distribuição de endereços IP pelo servidor DHCP para identificar os clientes e seus fluxos. A abordagem foi empregada para analisar o conjunto de dados de uso da rede sem fio institucional da Universidade Federal Fluminense e identificou 5 perfis distintos de uso da rede. Dentre os perfis identificados, é possível verificar que o perfil com melhor qualidade de experiência é associado a fluxos de usuários que estão conectados a pontos de acesso com menos de 25 usuários e, em até

90% dos casos, com menos de 15 usuários. O perfil de melhor qualidade de experiência alcança taxa de transmissão da ordem de 10 Mb/s, enquanto o perfil com menor qualidade de experiência apresenta taxa de transmissão bastante variável. O perfil com menor qualidade de experiência registra picos de mais de 50 clientes conectados em um mesmo ponto de acesso. Contudo, esse perfil se caracteriza por fluxos rápidos e com pouca transmissão de dados típico de aplicativos de mensagens instantâneas ou redes sociais. Como trabalhos futuros, pretende-se estender a proposta para o processamento *online* de fluxos para o cálculo de novos perfis e a adaptação de perfis existentes em tempo real.

Referências

[Andreoni Lopez et al., 2017] Andreoni Lopez, M., Silva, R. S., Alvarenga, I., Rebello, G., Sanz, I. J., Lobato, A., Mattos, D., Duarte, O. C. M. B. e Pujolle, G. (2017). Collecting and characterizing a real broadband access network traffic dataset. Em *2017 1st Cyber Security in Networking Conference (CSNet'17)*, Rio de Janeiro, Brazil.

[Balbi et al., 2012] Balbi, H., Fernandes, N., Souza, F., Carrano, R., Albuquerque, C., Muchaluat-Saade, D. e Magalhaes, L. (2012). Centralized channel allocation algorithm for ieee 802.11 networks. Em *2012 Global Information Infrastructure and Networking Symposium (GIIS)*, p. 1–7.

[Balbi et al., 2016] Balbi, H., Passos, D., Carrano, R., Magalhaes, L., e Albuquerque, C. (2016). Análise e solução para o problema da instabilidade de associação em redes IEEE 802.11 densas. Em *XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2016*.

[Biswas et al., 2015] Biswas, S., Bicket, J., Wong, E., Musaloiu-E, R., Bhartia, A. e Aguayo, D. (2015). Large-scale measurements of wireless network behavior. Em *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, p. 153–165, London, United Kingdom. ACM.

[Cisco, 2017] Cisco, V. N. I. (2017). Global mobile data traffic forecast update, 2016–2021 white paper. *Document ID*, 1454457600805266.

[Dely et al., 2012] Dely, P., Kassler, A., Bayer, N., Einsiedler, H. e Peylo, C. (2012). Optimization of wlan associations considering handover costs. *EURASIP Journal on Wireless Communications and Networking*, 2012(1):255.

[Divgi e Chlebus, 2013] Divgi, G. e Chlebus, E. (2013). Characterization of user activity and traffic in a commercial nationwide wi-fi hotspot network: global and individual metrics. *Wireless Networks*, 19(7):1783–1805.

[Ghosh et al., 2011] Ghosh, A., Jana, R., Ramaswami, V., Rowland, J. e Shankaranarayanan, N. K. (2011). Modeling and characterization of large-scale wi-fi traffic in public hot-spots. Em *2011 Proceedings IEEE INFOCOM*, p. 2921–2929.

[Guo et al., 2014] Guo, X., Chan, E. C. L., Liu, C., Wu, K., Liu, S. e Ni, L. M. (2014). Shopprofiler: Profiling shops with crowdsourcing data. Em *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, p. 1240–1248.

[Jang et al., 2017] Jang, R., Cho, D., Noh, Y. e Nyang, D. (2017). Rflow+: An sdn-based wlan monitoring and management framework. Em *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, p. 1–9.

[Joe-Wong et al., 2013] Joe-Wong, C., Sen, S. e Ha, S. (2013). Offering supplementary wireless technologies: Adoption behavior and offloading benefits. Em *2013 Proceedings IEEE INFOCOM*, p. 1061–1069.

[Li et al., 2013] Li, B., Springer, J., Bebis, G. e Gunes, M. H. (2013). A survey of network flow applications. *Journal of Network and Computer Applications*, 36(2):567 – 581.

[Magalhães e Mattos, 2018] Magalhães, L. C. S. e Mattos, D. M. F. (2018). Caracterização do uso de uma rede sem fio de grande porte distribuída por uma ampla Área. *XVII Workshop em Desempenho de Sistemas Computacionais e de Comunicação (WPerformance - CSBC 2018)*, 17(1/2018).

[Manweiler et al., 2013] Manweiler, J., Santhapuri, N., Choudhury, R. R. e Nelakuditi, S. (2013). Predicting length of stay at wifi hotspots. Em *Proceedings IEEE INFOCOM 2013*, p. 3102–3110.

[Mattos et al., 2019] Mattos, D. M. F., Velloso, P. B. e Duarte, O. C. M. B. (2019). An agile and effective network function virtualization infrastructure for the Internet of Things. *Journal of Internet Services and Applications*, 10(1):6.

[Oliveira et al., 2016] Oliveira, L., Obraczka, K. e Rodríguez, A. (2016). Characterizing user activity in wifi networks: University campus and urban area case studies. Em *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '16, p. 190–194, Malta. ACM.

[Qian et al., 2011] Qian, F., Wang, Z., Gerber, A., Mao, Z., Sen, S. e Spatscheck, O. (2011). Profiling resource usage for mobile applications: A cross-layer approach. Em *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, p. 321–334, Bethesda, Maryland, USA. ACM.

[Sen et al., 2013] Sen, S., Joe-Wong, C., Ha, S. e Chiang, M. (2013). A survey of smart data pricing: Past proposals, current plans, and future trends. *ACM Comput. Surv.*, 46(2):15:1–15:37.

[Shye et al., 2010] Shye, A., Scholbrock, B., Memik, G. e Dinda, P. A. (2010). Characterizing and modeling user activity on smartphones: Summary. Em *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '10, p. 375–376, New York, New York, USA. ACM.

[Wang et al., 2014] Wang, Y., Yang, J., Chen, Y., Liu, H., Gruteser, M. e Martin, R. P. (2014). Tracking human queues using single-point signal monitoring. Em *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, p. 42–54, New York, NY, USA. ACM.

[Xu et al., 2011] Xu, Q., Erman, J., Gerber, A., Mao, Z., Pang, J. e Venkataraman, S. (2011). Identifying diverse usage behaviors of smartphone apps. Em *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, p. 329–344, Berlin, Germany. ACM.

[Zhang et al., 2016] Zhang, X., Wang, C., Li, Z., Zhu, J., Shi, W. e Wang, Q. (2016). Exploring the sequential usage patterns of mobile internet services based on markov models. *Electronic Commerce Research and Applications*, 17:1 – 11.

Balanceamento de Tráfego entre Servidores de Vídeo MPEG-DASH em Redes Definidas por Software

Edenilson Jônatas dos Passos[1], **Adriano Fiorese**[1]

[1]Universidade do Estado de Santa Catarina (UDESC)
Campus Universitário Prof. Avelino Marcante - Rua Paulo Malschitzki, 200 - Zona Industrial Norte, Joinville - SC, 89219-710

edenilson.passos@yahoo.com, adriano.fiorese@udesc.br

Abstract. *With popularization of video streaming service in recent years, new video distribution technologies have been created. Currently, one of the most promising ones is the Moving Picture Expert Group Dynamic Adaptive Streaming over HTTP or MPEG-DASH. However, with the limitation of the TCP/IP network structure, the end user quality of experience (QoE) may be affected. One issue that can affect user QoE is the workload of content servers. Thus, the imbalancing of server's workload comprising user's attendance can lead to a content server provider non optimized choice. This work presents a load-balancing solution between MPEG-DASH video servers based on Software Defined Networks, using as a balancing workload metric the throughput of the content server.*

Resumo. *Com a popularização do serviço de streaming de vídeo nos últimos anos, novas tecnologias de distribuição de vídeo foram criadas. Uma das mais promissoras atualmente é a chamada de Moving Picture Expert Group Dynamic Adaptive Streaming over HTTP, ou MPEG-DASH. Contudo, com a limitação da estrutura de rede TCP/IP, a Qualidade de Experiência (QoE) do usuário final pode ser afetada. Um dos fatores que podem afetar a QoE do usuário é a carga de trabalho dos servidores de conteúdo. Assim, o desbalanceamento da carga de atendimento dos usuários entre os servidores pode gerar uma escolha não otimizada do provedor de conteúdo. Este trabalho apresenta uma solução de balanceamento de carga entre servidores de vídeo MPEG-DASH baseada em Redes Definidas por Software, utilizando como métrica de balanceamento a taxa de transferência do servidor de conteúdo.*

1. Introdução

A utilização da Internet na transmissão de vídeo possui papel relevante nos modelos de negócio dos provedores de conteúdo atuais. Isso é benéfico para os usuários pois possibilita um grande controle sobre quando e onde o conteúdo pode ser assistido. Uma pesquisa realizada pela empresa *The Nielsen Company* revelou que dentre um total de 30000 participantes de 61 países, 65% afirmam consumir algum tipo de programação de Vídeo Sob Demanda [Nielsen 2016]. Essa mesma pesquisa ainda revela a divisão total da atuação da Internet na distribuição de conteúdo de vídeo baseada em continentes. Na Ásia e Oceania, chega a 32%, na América, 35%, na África 21% e na América Latina 21%. Além disso, segundo [Cisco 2018], até o ano de 2022, é esperado que o tráfego de vídeo na Internet chegue a 82% do total.

Com a popularização de serviços de *streaming*, e consequentemente a sobrecarga nos enlaces bem como das restrições impostas a esse tipo de tráfego pelos provedores de serviço de acesso à Internet, empresas começaram a desenvolver formatos e conteúdo de vídeo adaptativo utilizando o protocolo HTTP como método de entrega. Contudo, com plataformas diferentes, existem configurações diferentes, o que gera a necessidade de padronização para que a interoperabilidade entre servidores e clientes de conteúdo de vídeo de diferentes fornecedores se torne possível [Sodagar 2011].

Assim, surgiu o padrão ISO/IEC 23009-1 ou Moving Picture Expert Group Dynamic Adaptive Streaming over HTTP (MPEG-DASH). O MPEG-DASH, foi projetado de maneira a balancear a qualidade de experiência do usuário (QoE) através de uma experiência fluída e com a melhor qualidade de vídeo possível [Duanmu et al. 2017]. Desse modo, o reprodutor de vídeo (*player*) MPEG-DASH no computador cliente se adapta de acordo com diversos fatores. Entre eles estão a resolução do vídeo, o *bitrate* de vídeo e de áudio, além da condição do *buffer* do *player*. Esses fatores variam de acordo com a largura de banda disponível, e a disposição geográfica dos servidores de conteúdo na *Content Distribution Network* (CDN) [Stockhammer 2011].

A CDN é responsável pelo armazenamento do conteúdo buscado pelos clientes e também pelo roteamento das requisições de conteúdo para um servidor mais próximo do cliente. Além disso, uma CDN permite o atendimento da demanda crescente dos usuários por conteúdo assumindo as responsabilidades pela manutenção da qualidade do serviço de distribuição dos conteúdos e sua escalabilidade. Entre essas responsabilidades encontra-se o balanceamento de carga [Bourke 2001], [Cardellini et al. 1999]. De acordo com [Doshi et al. 2015], o caso de atraso que consequentemente mais afeta a QoE do usuário é o congestionamento do servidor, pois é percebido pelo usuário como interrupção e ruído na imagem e áudio do conteúdo.

Uma abordagem contemporânea para o balanceamento de carga em CDNs para distribuição de conteúdo de vídeo é baseada no atendimento das requisições dos diversos clientes por vários servidores distribuídos na CDN. Para tal, a atribuição de qual servidor atenderá qual cliente é feita através da implantação personalizada de servidores *Domain Name Server* (DNS) por provedor de CDN. As entradas no DNS então, são adaptadas a todo momento de modo a fornecer o balanceamento de carga entre os servidores. Contudo, fluxos de grande volume e de longa duração, como por exemplo o tráfego de vídeo sob demanda (VoD), dificilmente são gerenciáveis através dessa abordagem DNS. A introdução de SDN é promissora para resolver este problema, pois instanciações de SDN como o *OpenFlow* permitem o redirecionamento transparente de fluxos na rede [Wichtlhuber et al. 2015], sem a necessidade de manipulação de serviços de infraestrutura.

Portanto, o problema do balanceamento de carga em CDNs, ou seja, entre os servidores de conteúdo, nesse caso particular de servidores de vídeo MPEG-DASH, sofre de dificuldades de gerenciamento com as abordagens tradicionais atuais. A contribuição deste artigo visa disponibilizar uma abordagem arquitetural baseada em SDN para a solução desse problema.

Sendo assim, o presente artigo trata da proposição de uma arquitetura para o balanceamento de carga em rede de distribuição de vídeo com o auxílio de SDN para realizar

a manipulação de fluxos necessários por meio do protocolo Openflow versão 1.3 e do controlador Ryu. O balanceamento executado tem como objetivo o alívio da carga de trabalho dos servidores origem do conteúdo e utiliza a vazão de dados dos servidores como métrica de balanceamento.

Este trabalho está organizado da seguinte maneira. Na Seção 2, os trabalhos relacionados são discutidos. Em seguida, na Seção 3, a arquitetura da solução proposta é detalhada. A Seção 4, apresenta a avaliação da proposta e a Seção 5 apresenta as considerações finais deste trabalho.

2. Trabalhos Relacionados

Em [Jiang et al. 2015] é apresentado um algoritmo adaptado de Dijkstra estendido para balanceamento de carga em uma CDN utilizando a tecnologia SDN. Segundo a justificativa utilizada, algoritmos que se utilizam de abordagens de seleção aleatória ou ainda a abordagem *Round Robin*, ao selecionar uma rota para o servidor de conteúdo, podem selecionar a rota mais longa (em relação a saltos) e / ou com congestionamentos, diminuindo drasticamente o desempenho do serviço. Com o algoritmo proposto, a intenção é selecionar o caminho mais curto e com menor congestionamento. Assim, são realizados diversos testes referentes a latência do servidor, tempo de resposta e *throughput*. Nesse trabalho, os resultados do algoritmo proposto são comparados com aqueles das abordagens de escolha aleatória e Round Robin.

Semelhante a [Jiang et al. 2015], a ideia central de [Kaysudu et al. 2016], é propor um algoritmo, em uma rede SDN, que encontra uma rota mais apropriada para se alcançar o melhor servidor de distribuição de conteúdo. De acordo com os estudos apresentados, o desempenho dos sistemas de transmissão de vídeo em execução na rede SDN pode ser aprimorado configurando as rotas pelas quais os pacotes de vídeo são transmitidos para os clientes com base em medições de largura de banda em tempo real.

Um trabalho que utiliza da área de inteligência artificial para balanceamento de carga foi abordado em [Chen-xiao and Ya-bin 2016]. Nele foi utilizada a abordagem SDN para fazer o redirecionamento dos fluxos de video conforme o resultado do algoritmo de balanceamento de carga proposto. Esse algoritmo é baseado em uma rede neural que utiliza como principais métricas a utilização da largura de banda, a taxa de pacotes perdidos, a latência e o número de saltos. Contudo, não tinha como foco redes de distribuição de conteúdo, sendo o principal objetivo do trabalho economizar a largura de banda através da seleção do melhor caminho possível sem congestionamento.

Um exemplo de trabalho que utiliza balanceamento de carga baseado em métricas para escolha do servidor mais adequado é [Carter and Crovella 1997]. Nele, os autores utilizaram ferramentas específicas para medir a latência e a largura de banda disponível do servidor para o cliente. Com essas métricas o algoritmo tomava as devidas decisões a respeito de qual servidor era o mais adequado para transmitir os dados. Entretanto, não foi utilizada da tecnologia VoD e nem do paradigma SDN.

Sendo assim, observando os mais recentes trabalhos que tratam do balanceamento de carga em sistemas de distribuição de conteúdo (especialmente vídeo), é possível notar possibilidade de melhorias. Especialmente, levando em conta a utilização do paradigma SDN para o estabelecimento do balanceamento de carga entre servidores de conteúdo,

executado diretamente em nível da infraestrutura de rede (camadas 2,3 e 4). Além disso, a utilização da métrica de vazão dos servidores envolvidos, bem como o monitoramento e utilização sob demanda desse valor torna o balanceamento de carga altamente dinâmico e escalável, seja na adição de clientes quanto de servidores.

3. Solução Proposta

Esta seção apresenta a abordagem proposta para o balanceamento de carga entre servidores de conteúdo MPEG-DASH. Tal abordagem contempla uma arquitetura de software implementada junto ao controlador SDN, envolvendo passos e ações específicas a serem executadas para o correto redirecionamento das requisições de vídeo do usuário. Estes redirecionamentos, em última análise, executam o balanceamento de carga de acordo com a métrica utilizada.

3.1. Arquitetura Proposta

Na arquitetura proposta, o controlador Ryu é o principal atuador do sistema. Ele é quem executa as principais funções do sistema, como a obtenção das métricas de taxa de transferência dos servidores e toma a decisão de qual é o mais adequado para então poder gerar e instalar as regras de redirecionamento de tráfego no *switch* OpenFlow. A abordagem adotada para a atuação do controlador, na arquitetura proposta, é reativa. Assim, uma vez que o cliente tente acessar o sítio web contendo a página do conteúdo de vídeo, esse pacote de dados da requisição é encaminhado ao controlador, caso no *switch* OpenFlow não haja regra correspondente.

A Figura 1 apresenta a arquitetura da solução proposta, destacando as atividades executadas e suas interações como aplicação pertencente ao controlador. Além disso, a Figura 1 indica a sequência numérica de execução dessas atividades como uma sequência de eventos.

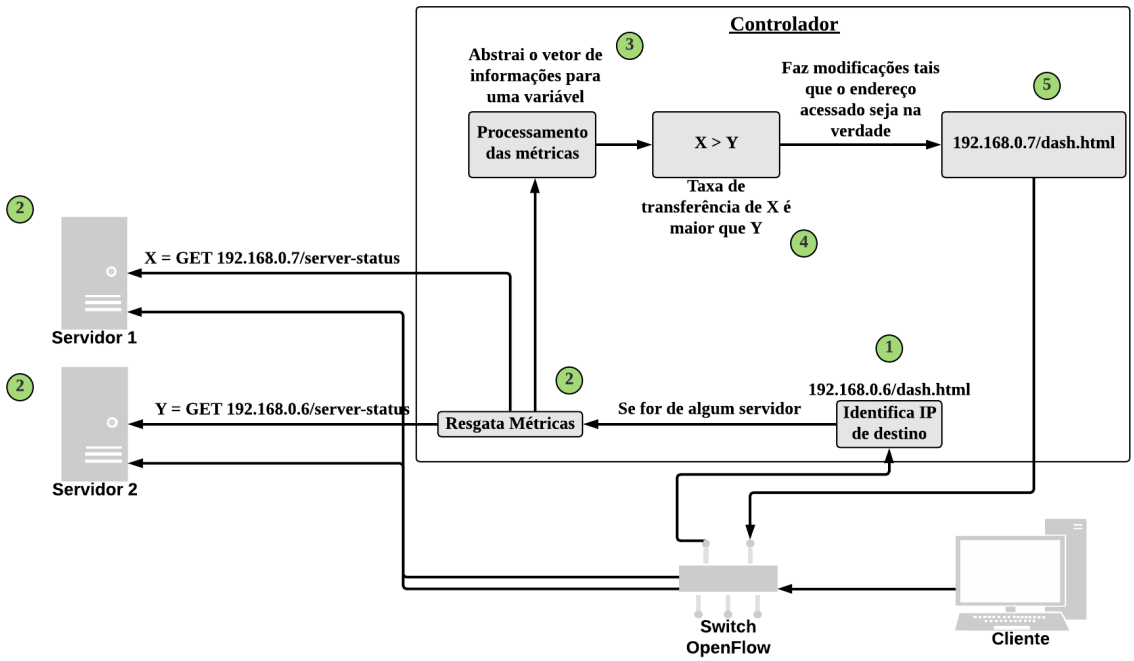


Figura 1. Arquitetura de balanceamento de carga proposta

Assim, de acordo com a arquitetura proposta, o pacote de dados representando a requisição do cliente, ao chegar ao *switch* OpenFlow, com o seu direcionamento ao controlador, põe em marcha a identificação do cliente requisitante e o conteúdo a ser entregue. Para tal, por meio do estabelecimento da conexão TCP e execução da requisição HTTP do conteúdo, uma função identifica o endereço IP de origem (1). Com o endereço de origem, o próximo passo é verificar o endereço de destino (1). Caso o endereço destino seja de um dos servidores de conteúdo conhecidos e disponíveis, o próximo passo compreendendo a obtenção das métricas desses servidores, é iniciado (2). A obtenção dos valores da taxa de transferência dos servidores é realizada pela aplicação de balanceamento desenvolvida junto ao controlador, obtida uma única vez durante a execução do vídeo, logo após o processo de obtenção do IP de origem. Ou seja, a cada nova conexão TCP endereçada a um dos servidores, os valores da métrica utilizada são resgatados. Portanto, o balanceamento de carga ocorre apenas em função do estabelecimento da conexão (e consequente sessão HTTP).

Após a obtenção dos valores da métrica utilizada, a aplicação de balanceamento desenvolvido junto ao controlador avalia qual servidor possui a maior taxa de transferência (3,4), presumindo que aquele que possui maior taxa de transferência está apto a receber mais conexões e propenso a menor latência. O próximo e último passo é a geração e instalação das regras no *switch* OpenFlow referentes aos endereços IP e MAC e à porta de destino do tráfego de conteúdo com base no melhor servidor escolhido (5).

A Figura 2 ilustra a sequência de eventos do sistema proposto. Primeiramente, o cliente faz a requisição do conteúdo. Essa requisição passa então pelo *switch* OpenFlow que a reencaminha ao controlador para tratamento e execução das medidas cabíveis. Assim, o controlador solicita aos servidores disponíveis a métrica de taxa de transferência. Com a métrica de todos os servidores, é feita a tomada de decisão de atendimento pelo que possui a taxa mais alta. Após isso, são geradas e executada a instalação das regras de fluxo no *switch*. Assim, com as regras instaladas, é de fato realizada a requisição ao servidor escolhido e este por sua vez retorna o conteúdo ao cliente utilizando-se das regras recentemente instaladas no *switch*.

3.2. Implementação

Para a instanciação da arquitetura proposta, além da implementação da aplicação de balanceamento de carga junto ao controlador Ryu, na falta de uma CDN para testes, uma infraestrutura que simule o acesso de clientes a vários servidores de conteúdo, em condições manipuláveis da rede, é necessária. Nesse sentido, o cenário de implementação adotado tem como elementos n servidores de conteúdo, um controlador, um *switch* OpenFlow e n clientes conforme Figura 3.

Assim, para o caso particular deste trabalho, devido a limitação de hardware, porém sem perda da objetividade, apenas dois servidores de conteúdo MPEG-DASH foram criados, ambos idênticos. Isto é, ambos possuem o mesmo conteúdo a ser disponibilizado. Com relação aos clientes, mesmo estando na rede Mininet, e portanto virtualizados, foi disponibilizado a eles acesso externo e portanto à Internet para que acessassem os servidores de conteúdo.

O cenário proposto baseado em rede foi elaborado de modo que a rede criada pelo emulador Mininet consiga acessar a rede LAN física (representando a rede WAN em

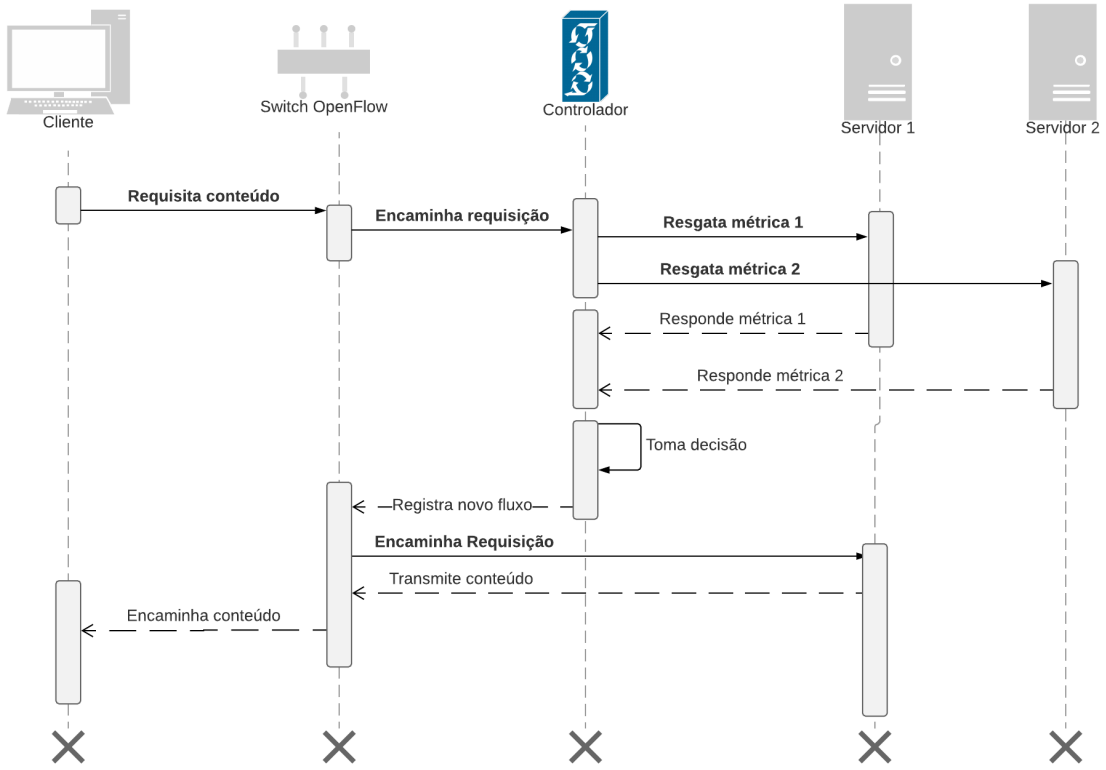


Figura 2. Sequência de eventos da solicitação do conteúdo vídeo

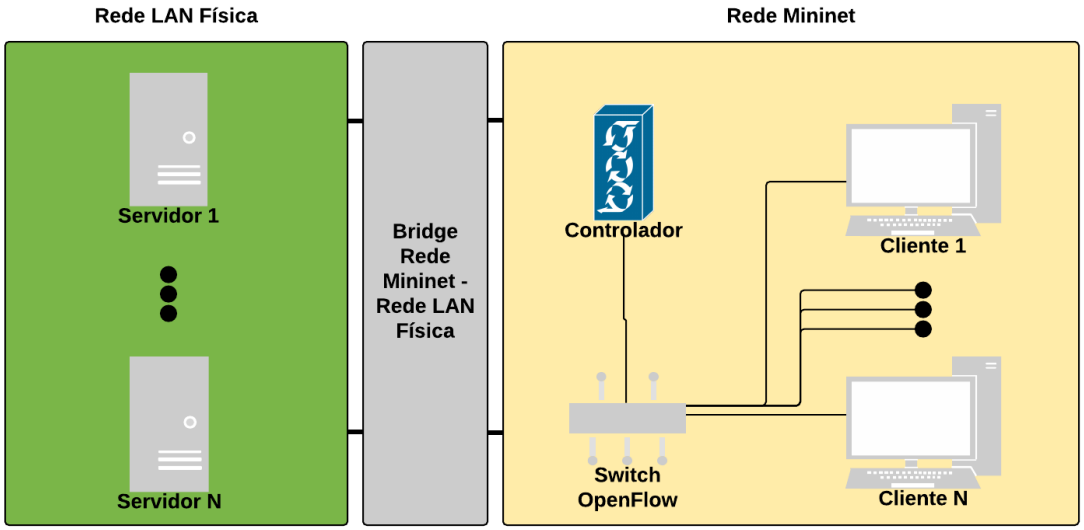


Figura 3. Cenário de implementação baseado em redes distintas

cenário real), na qual os servidores estão hospedados. Essa conexão ocorre por meio de uma ponte (*bridge*) que conecta essas duas interfaces através do *switch* OVS.

Mais especificamente, é realizada a configuração dos clientes (consumidores de conteúdo de vídeo MPEG-DASH) virtualizados, existentes na rede Mininet, de forma

que os mesmo recebam endereços IP da rede LAN física real que tem acesso a Internet, conforme Figura 4. Assim, no passo 1, ao ser inicializado, o cliente recebe endereço IP aleatório associado a rede interna Mininet (*h1-eth0*). No passo 2, é executado o comando para desassociação desse IP (*ifconfig h1-eth0 0*). Logo, esse cliente (*host*) não possui endereço IP. No passo 3, é adicionada a interface de rede do equipamento físico que está executando o programa Mininet ao *switch* OVS utilizado pela rede Mininet (*sudo ovs-vsctl add-port s1 enp0s3*). Desse modo, todos os dispositivos conectados a esse *switch* (*hosts* na rede Mininet) terão acesso a Internet quando feita a nova associação de IP no passo 4. Essa nova associação, se dá agora pelo serviço de DHCP (*Dynamic Host Configuration Protocol*) executando no dispositivo físico que executa o Mininet e a rede emulada por ele.

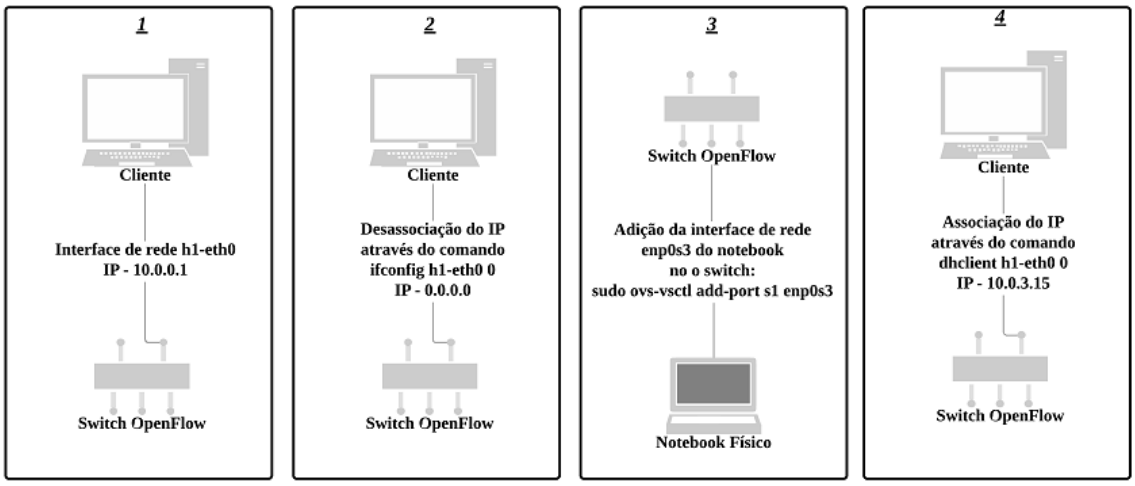


Figura 4. Passo a passo configuração IP dos *hosts*

Para simular um cenário mais realista em termos de consumo de vídeo MPEG-DASH, foi idealizado que todos os *hosts* (clientes) criados dentro da rede Mininet possam utilizar um navegador de Internet para executar a visualização do conteúdo recebido. Nesse caso, o navegador Google Chrome, com alterações para execução em modo administrador, foi utilizado pois permite a utilização facilitada de *player* que permite a visualização de conteúdo MPEG-DASH.

A instalação do *player* MPEG-DASH é um passo importante na implementação do trabalho. Isso se deve ao fato de que reprodutores de vídeo/áudio convencionais não tem suporte para reprodução de tal padrão.

O *player Dash.js* é, de fato, um arquivo em *javascript* que pode ser descarregado pelo sítio web de origem [dash.js 2012]. Sua instalação se dá pela inclusão desse pacote (arquivo) no cabeçalho (*header*) da página web que disponibiliza o reprodutor ao usuário.

Para o resgate dos valores da métrica taxa de transferência nos servidores de vídeo MPEG-DASH, o servidor Apache fornece uma ferramenta conhecida como *server-status* que ativa uma página HTML (*Hyper Text Markup Language*) com todas as possíveis informações e métricas de desempenho do servidor. Essa página pode ser acessada remotamente. Assim, o resgate, executado pela aplicação de balanceamento de tráfego desenvolvida junto ao controlador Ryu, dos valores da métrica taxa de transferência érealizado

por meio da requisição HTTP GET, dessa página, nos servidores designados. Portanto, quando retornada tal página web, é executado o tratamento dos dados recebidos para que o valor da taxa de transferência seja o único a ser comparado. Um ponto importante a ser destacado é que, para que o *status* do servidor possa ser acessado, nas configurações do servidor Apache (*apache.conf*), devem ser adicionados os endereços que têm acesso ao servidor, pois caso contrário a requisição será negada.

A elaboração do algoritmo balanceador de carga proposto foi realizada com base na aplicação de *switch* camada 2 da biblioteca disponível do controlador Ryu. A partir dessa aplicação, foram executadas as modificações necessárias de tal modo que quando uma requisição de conteúdo MPEG-DASH é recebida no *switch* OpenFlow, todos os pacotes TCP enviados para o IP do servidor de conteúdo destinatário que pretendem iniciar uma conexão TCP, não vão ser de fato entregues diretamente ao servidor, mas redirecionados para o controlador para processamento.

A aplicação de balanceamento de carga então requisita os dados de taxa de transferência (por meio da página HTML de *status*) dos servidores através de requisições HTTP GET. Após receber a resposta, e executar o tratamento para encontrar os valores da métrica taxa de transferência, ela poderá decidir qual servidor real é o mais adequado para entregar o conteúdo. Em seguida, a aplicação de balanceamento de carga modifica o plano de dados da rede instalando regras de fluxo cuja ação deve reescrever o endereço IP, MAC e a porta de destino de todos os pacotes futuros do cliente para o endereço IP, MAC e porta do servidor escolhido. O mesmo deve ser realizado no sentido contrário, isto é, do servidor para o cliente. Isso deve ocorrer para que o cliente não perceba que o endereço IP do servidor foi redirecionado.

Portanto, é esperado que o usuário não perceba qualquer tipo de mudança na reprodução do conteúdo. Caso o cliente (*player*) MPEG-DASH executando no navegador Google Chrome utilize, por exemplo, o endereço 10.0.0.1 para acessar o servidor de conteúdo, mas o controlador decidir que o servidor 10.0.0.2 é mais adequado, o redirecionamento será executado automaticamente e o usuário/cliente não saberá que essa troca foi feita.

4. Experimentos e Resultados

Todos os experimentos idealizados tem como principal objetivo observar o comportamento da reprodução do vídeo no computador do usuário final. Deste modo, todas as métricas de avaliação do balanceamento de carga proposto para as quais os experimentos foram elaborados, tem relação direta ou indireta com a qualidade de experiência do usuário final.

4.1. Ambiente de Testes

Todos os experimentos foram realizados em máquinas virtuais com o auxílio da ferramenta VirtualBox. O primeiro dispositivo utilizado para executar a máquina virtual é um computador de mesa. Este tem como papel hospedar a rede Mininet, contendo os *hosts*, o *switch* OpenFlow (OpenVSwitch), o controlador e o servidor de conteúdo que está separado do equipamento que contém o ambiente virtual Mininet. O sistema operacional utilizado foi o Ubuntu com 3 GB de RAM com processador de 4 núcleos a 3.5 GHz. O segundo dispositivo utilizado é um notebook. As especificações da máquina virtual são

semelhantes as anteriores. Porém, por dispor de um processador inferior executando a apenas 3.2 GHz, foi elencado somente como servidor de conteúdo.

O *player* MPEG-DASH utilizado é o *dash.js* por ser de código aberto. Logo, é possível realizar modificações conforme necessário. Particularmente, modificações (funções) foram implementadas para permitir boa parte da análise da qualidade de experiência do usuário, ou seja, permitir acompanhar o comportamento de certas estatísticas de transmissão do vídeo durante a reprodução do conteúdo. Entre as funções desenvolvidas estão:

- Tempo para aparição do primeiro quadro de vídeo no reprodutor. Como discutido anteriormente, se trata de uma métrica de QoE.
- A cada segundo é impresso o *bitrate* atual do vídeo e áudio, juntamente com a respectiva resolução.
- A cada 15 segundos é feito um teste de velocidade da largura de banda.
- Opcionalmente, no início do vídeo é possível ativar uma função que mostra todas as qualidades disponíveis do vídeo e áudio do conteúdo (resolução, *bitrate* de áudio e vídeo).

Para criar a rede no Mininet criam-se inicialmente os *hosts* (clientes) que executarão os navegadores e por conseguinte o *player* MPEG-DASH. Cria-se em seguida o *switch* virtual com o open vswitch e conectam-se cada um desses *hosts* às portas do *switch*. Após as devidas configurações, o cliente (navegador executando no *host* do Mininet) está pronto para acessar o conteúdo disponibilizado na página do servidor Apache.

Com relação aos conteúdos de vídeo/áudio disponibilizados, foram escolhidos dois *datasets* de vídeos de licença *Creative Commons* (CC) oferecidos para testes pelo YouTube. O primeiro *dataset* é chamado de ***CAR CENC***. O vídeo possui 181 segundos de duração, e é disponibilizado em 6 qualidades (diferentes resoluções) de vídeo com variação de 256×144 à 1920×1080 pixels, e uma faixa de áudio. Nesse caso, tanto o vídeo quanto o áudio estão disponibilizados em segmentos (chunks) de 2 segundos.

O segundo *dataset* é chamado de ***FEELINGS VP9***. Este *dataset* possui compressão de vídeo pelo método VP9 e por isso, seu tamanho é consideravelmente menor em relação ao anterior. Porém, apresenta como qualidade máxima somente alta definição, ou seja, não é disponibilizado *Full HD*. O vídeo possui 136 segundos de duração, e é disponibilizado em 6 qualidades de vídeo com variação de 426×240 à 1280×720 pixels.[1] e uma faixa de áudio.

4.2. Avaliação de QoE

Para além da abordagem de balanceamento de carga orientada a tráfego proposta, baseada na métrica taxa de transferência, outras duas abordagens foram desenvolvidas. Uma delas utiliza um algoritmo de escolha aleatória do servidor e a outra, o algoritmo *Round Robin* entre os servidores de conteúdo MPEG-DASH. O algoritmo de escolha aleatória foi desenvolvido de maneira simplista, como aplicação do controlador. Assim, a cada nova conexão tratada pela aplicação de balanceamento junto ao controlador, duas variáveis, cada uma representando um servidor, recebem um número aleatório de 0 a 1000. Aquela

[1]Nos resultados, quando se falar em *dataset carro*, significa que trata-se do dataset ***CAR CENC***. Já o ***FEELINGS VP9*** é lido como *paraquedas*

que apresentar o maior número aleatório, é o servidor escolhido. Já para a abordagem Round Robin, foi criada uma variável global que é incrementada em um a cada nova conexão tratada. Como os testes foram realizados contando apenas com dois servidores de conteúdo, quando o número for par o servidor 1 é escolhido. Consequentemente, o servidor 2 é escolhido caso o número seja impar. É importante ressaltar que para a realização dos experimentos com as abordagens sem a utilização de SDN, a taxa de transferência foi monitorada manualmente, para que quando houvesse a requisição de conexão, ela fosse feita para o servidor com a menor taxa de transferência. A intenção é demonstrar que existirão atrasos caso a taxa de transferência for baixa.

Os resultados a seguir apresentados foram obtidos da seguinte maneira. No total, foram realizados 10 testes para cada *dataset* de cada método/abordagem proposta sendo que o valor da média dessas execuções é usado como resposta. A cada teste, o controlador era encerrado assim como a rede Mininet e o *cache* do navegador era esvaziado.

As métricas para avaliação dos da abordagem proposta por meio dos experimentos realizados estão relacionadas com a mensuração da QoE do cliente, e são:

- A média do *bitrate* para cada segundo de execução de vídeo. Assim, quanto maior a taxa, melhor a qualidade do vídeo exibido. Para tal, foram criados dois gráficos, um para cada *dataset*, com a média geral de todos os métodos envolvidos.
- O número de vezes que a qualidade do vídeo mudou. Quanto menor esse número, melhor a qualidade de experiência do usuário final.
- O tempo de carregamento da página. Quanto menor, melhor.
- O tempo para a aparição do primeiro quadro do vídeo no *player*. Quanto menor, melhor.
- O número de congelamentos (*stalls*) no vídeo. Quanto menor melhor. O ideal é zero.
- A média de largura de banda utilizada pelo cliente.

Assim, foram executados os experimentos de requisição e reprodução dos vídeos dos *datasets* utilizados, que capturaram os valores para as métricas elencadas anteriormente. A Tabela 1 apresenta os resultados obtidos. Assim, é possível notar que para a métrica tempo de carregamento da página, a abordagem proposta é mais eficiente do que todas as outras abordagens quando o *dataset* é o ***FEELINGS VP9*** (paraquedas). Em relação ao outro *dataset*, ele é inferior somente a abordagem sem SDN. Uma das possíveis justificativas é que na abordagem sem SDN, não existe escolha de servidor baseado em métrica e processamento e portanto, tende a ser mais ágil.

A mesma situação ocorre com a terceira métrica de avaliação: o tempo de espera para o primeiro *frame* do vídeo ser visualizado. O método proposto é mais eficiente no *dataset* do paraquedas. Contudo, perde para o método sem SDN no *dataset* carro. A justificativa para tal, é idêntica a da métrica anterior.

Na quarta métrica, foi contabilizado o número de vezes em que a qualidade do vídeo foi alterada durante a reprodução no cliente. Antes de tudo, é válido ressaltar que é esperado que o vídeo tenha um número mínimo de troca de qualidades. Nesse caso, essa quantidade é 3. Isso se deve ao fato de que, quando o vídeo é solicitado, a transmissão é iniciada em qualidade inferior, geralmente a mínima no primeiro *frame* pois o *player* precisa se certificar que existe de fato largura de banda suficiente para aumentar a qualidade. Por isso, geralmente são notadas 3 trocas de qualidade nos segundos iniciais.

	Sem SDN		Randômico		Round Robin		Abordagem Proposta	
	Carro	Paraquedas	Carro	Paraquedas	Carro	Paraquedas	Carro	Paraquedas
Carregar pagina (ms)	574,30 σ = 47,46	542 σ = 37,38	819,90 σ = 243,78	819,10 σ = 298,97	987,10 σ = 227,51	1066,40 σ = 49,84	679,80 σ = 354,21	525,80 σ = 450,55
Primeiro frame (ms)	1078 σ = 55,81	1144,10 σ = 118,53	1371,80 σ = 402,50	1242,30 σ = 312,76	1486,20 σ = 278,96	1679,40 σ = 402,50	1141 σ = 410,58	1011,80 σ = 582,86
Trocas de qualidade	5,3 σ = 2,86	3 σ = 0	3,7 σ = 2,14	3 σ = 0	3,6 σ = 1,01	3 σ = 0	3,5 σ = 1,36	3 σ = 0
Stalls	0,8 σ = 0,97	0	1 σ = 1,26	0	0	0,2 σ = 0,6	0,6 σ = 0,91	0,4 σ = 0,8
Largura de banda (Mbps)	2,39 σ = 0,26	0,30 σ = 0,09	2,11 σ = 1,36	1,08 σ = 1,29	3,88 σ = 0,24	2,41 σ = 0,10	3,47 σ = 0,59	2,39 σ = 0,15

Tabela 1. Tabela de resultados finais

Como o segundo *dataset* é mais "leve", todos os métodos implementados conseguiram o melhor resultado (3 mudanças de qualidade). No *dataset* carro, no entanto, sendo um vídeo de qualidade máxima de 1920 $\times$ 1080 *pixels*, sem compressão, portanto mais "pesado", houve mais trocas de qualidade em todos os métodos. Isso ocorre pois esse vídeo em sua qualidade máxima exige muita largura de banda, e conforme o funcionamento do *player* MPEG-DASH, quando não há largura de banda suficiente, a qualidade dos segmentos de vídeo requisitados é diminuída. Em alguns casos, em especial no método sem SDN, há mais trocas de qualidade principalmente ao meio da reprodução, o que afeta negativamente a qualidade de experiência do usuário. Uma das explicações para tal, se deve ao fato de que os testes foram realizados com o limite de largura de banda imposto pelo programa *Traffic Monitor* e em alguns momentos houve instabilidade da largura de banda, forçando o *player* a diminuir a qualidade requisitada dos vídeos.

Em relação a quinta métrica, o número de *stalls*, é importante ressaltar que um *stall* geralmente ocorre quando não há largura de banda suficiente para consumir o vídeo num determinado espaço de tempo e o *player* não consegue diminuir a qualidade de vídeo em tempo ótimo para que a reprodução continue sem interrupções. Os resultados obtidos foram positivos em todos os testes, isto é, houve nenhum ou baixo número de *stalls*. Embora em algumas situações ocorreram de fato *stalls*, a ocorrência deles foi no início do vídeo (aos zero segundos) de vídeo, logo, era pouco perceptível ao usuário. Esse comportamento de *stall* no segundo zero do vídeo se deve ao comportamento escada que o *player* utiliza para reprodução, especialmente nos segundos iniciais. Por conta disso, pouco afetou a qualidade de experiência. O melhor resultado foi da abordagem *Round Robin* com nenhum *stall* no *dataset* carro e uma média de 0,2 no *dataset* paraquedas. Por outro lado, o pior resultado foi da abordagem aleatória. Com pelo menos um *stall* por reprodução no *dataset* carro.

Por último, o consumo de largura de banda durante a reprodução do vídeo pode ser observado. A abordagem sem SDN se mostrou superior nessa métrica no *dataset* do paraquedas com um consumo médio de apenas 0,3 Mbps. Já o método proposto, é o

segundo mais custoso, com 2,39 Mbps sendo utilizados.

Foi realizado também, o monitoramento da variação do *bitrate* do vídeo em relação ao tempo. Para tal, o vídeo é executado e consumido por inteiro. Haja vista as funcionalidades de monitoramento implementadas no *player*, neste caso em especial, a de monitoramento do *bitrate*, a cada execução é gerado um relatório com todas as características durante a execução. Após 10 execuções para cada *dataset* e para cada abordagem, é executado o cálculo da média de *bitrate* em cada segundo de execução. Por isso, em algumas situações nos gráficos apresentados nas Figuras 5 e 6 o *bitrate* correspondente a abordagens diferentes, encontra-se sobreposto.

De qualquer forma, ao observar a Figura 5 nota-se que dentre as abordagens testadas, a que obteve a menor média de *bitrate* foi a sem SDN. As outras abordagens, contudo, foram bastante similares. No início, do segundo 1 ao 10, todos tem comportamento semelhante devido ao efeito escada comentado anteriormente. Já a partir do segundo 10 é notável uma leve variação entre eles. Entretanto, do segundo 40 ao segundo 70, é possível notar uma diminuição expressiva da média do *bitrate* da abordagem sem SDN.

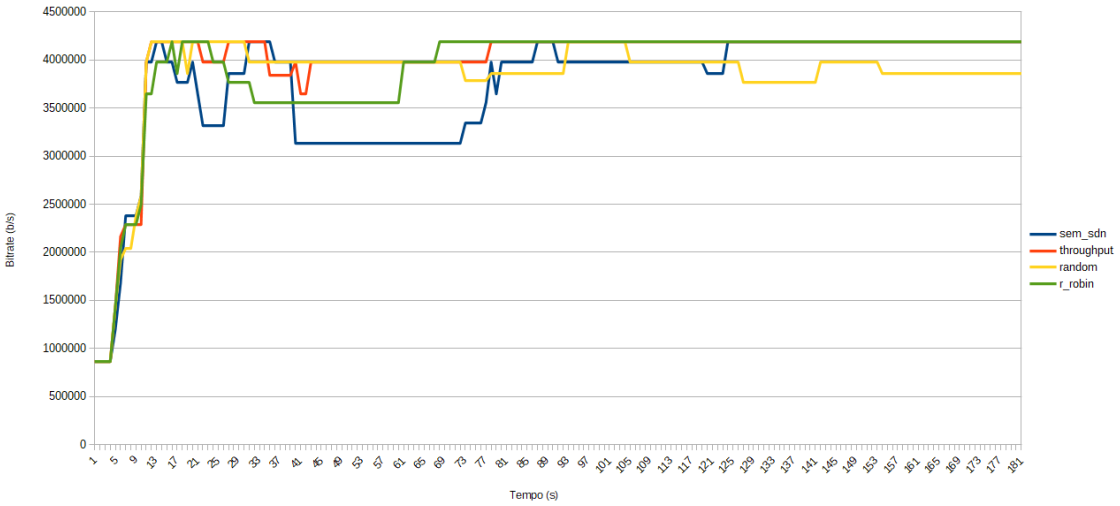


Figura 5. Resultado Final, média bitrate por método - Carro

Já no segundo *dataset* (paraquedas), conforme a Figura 6, o comportamento de todas as abordagens foi extremamente parecido. Isso se deve ao fato de que esse *dataset* era um vídeo cuja resolução mais alta é de 1280 $\times$ 720 *pixels*, menor portanto que o *dataset* carro, além da compressão VP9. Por essas razões o descarregamento desse vídeo era rápido.

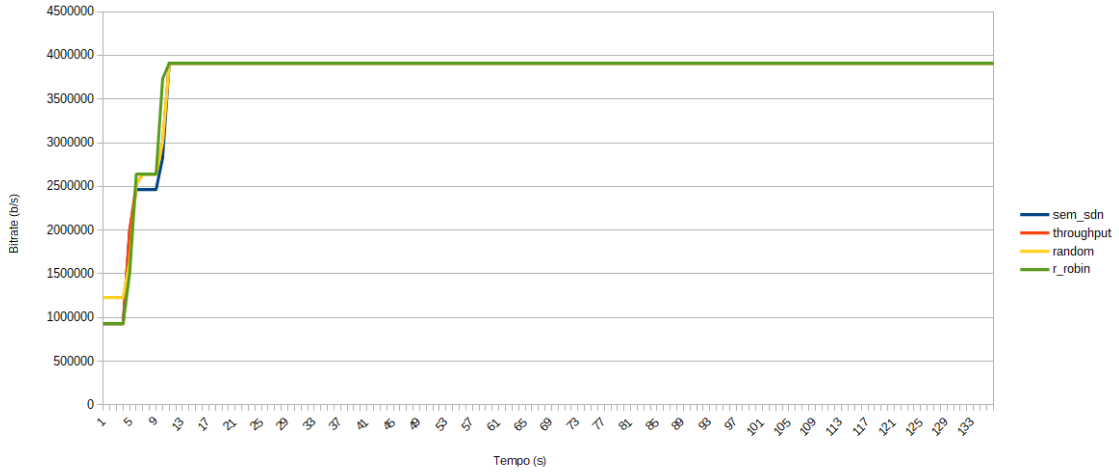


Figura 6. Resultado Final, média bitrate por método - Paraquedas

De modo geral, de acordo com os experimentos realizados, o método proposto se mostra promissor, haja visto que consegue diminuir o tempo de resposta da requisição de conteúdo de vídeo MPEG-DASH, mantendo média de *bitrate* elevada, preservando assim a qualidade de experiência do usuário. Entretanto, mesmo com desempenho elevado é o segundo método que consome mais largura de banda, o que é um fator importante a ser levado em consideração caso seja implementado em produção.

5. Conclusão

Com a popularização da Internet e avanços tecnológicos referentes a arquitetura e estrutura de rede, o consumo de vídeo se tornou uma prática diária no cotidiano de muitas pessoas. Contudo, apesar desses avanços e constantes melhorias, ainda há diversos desafios a serem superados. Um deles é o balanceamento de carga de modo que a rede de distribuição de conteúdo consiga suportar uma alta demanda de requisições e não prejudique a qualidade de experiência do usuário final.

Este trabalho apresenta uma possível solução ou pelo menos atenuação a esse problema. Para tal, foi idealizado um sistema que utiliza o paradigma Redes Definidas por Software para interceptar os pacotes durante a reprodução do conteúdo em vídeo, e através da análise da métrica taxa de transferência dos servidores de conteúdo disponíveis, efetuar a mais adequada escolha de servidor, para servir o conteúdo, de maneira a maximizar sua taxa de transferência. A avaliação realizada teve como principal objetivo destacar que apesar de se tratar de uma solução que modifica os fluxos de conexão, não há percepção desse redirecionamento na reprodução do conteúdo consumido pelo usuário e portanto da qualidade de experiência vivenciada. Além disso, a abordagem proposta apresenta ganhos em diversos aspectos como tempo de resposta, número de trocas de qualidade e *stalls*.

Contudo, são necessários mais experimentos referentes a escalabilidade horizontal da abordagem proposta para a constatação de sua completa aplicabilidade. Ainda, é intencionado realizar comparações com outras abordagens, bem como incrementar a decisão de balanceamento agregando valores de mais métricas relevantes tanto à QoE do usuário final, como de ganho em melhor aproveitamento dos recursos computacionais dos servidores de vídeo.

Referências

Bourke, T. (2001). *Server load balancing*. O'Reilly, 1st edition.

Cardellini, V., Colajanni, M., and Yu, P. (1999). Dynamic load balancing on web-server systems. *IEEE Internet Computing*, 3(3):28–39.

Carter, R. L. and Crovella, M. E. (1997). Server selection using dynamic path characterization in wide-area networks. In *Proceedings of INFOCOM '97*, volume 3, pages 1014–1021 vol.3.

Chen-xiao, C. and Ya-bin, X. (2016). Research on load balance method in SDN. *International Journal of Grid and Distributed Computing*, 9(1):25–36.

Cisco (2018). Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper.

dash.js (2012). A reference client implementation for the playback of MPEG DASH via javascript and compliant browsers.: Dash-industry-forum/dash.js. Disponível em `https://github.com/Dash-Industry-Forum/dash.js`. Acesso 13 Out. de 2018.

Doshi, B., Kumar, C., Piyush, P., and Vutukuru, M. (2015). Webq: A virtual queue for improving user experience during web server overload. In *2015 IEEE 23rd International Symposium on Quality of Service (IWQoS)*, pages 135–140.

Duanmu, Z., Ma, K., and Wang, Z. (2017). Quality-of-experience of adaptive video streaming: Exploring the space of adaptations. In *MM '17 Proceedings of the 2017 ACM on Multimedia Conference*, pages 1752–1760. ACM Press.

Jiang, J.-R., Yahya, W., and Ananta, M. (2015). Load balancing and multicasting using the extended dijkstra's algorithm in software defined networking. *Frontiers in Artificial Intelligence and Applications*, 274:2123–2132.

Kaysudu, E., Çetinkaya, C., Hergüner, K., and Sayıt, M. (2016). Server selection for video streaming applications over software defined networks. In *2016 24th Signal Processing and Communication Application Conference (SIU)*, pages 1965–1968.

Nielsen, T. C. (2016). How worldwide viewing habits are changing in the evolving media landscape. `http://www.nielsen.com/content/dam/nielsenglobal/eu/docs/pdf/Nielsen-global-video-on-demand.pdf`, 73, Avenue des Pléiades B-1200 Brussels. Acesso 15 Ago. de 2018.

Sodagar, I. (2011). The mpeg-dash standard for multimedia streaming over the internet. *IEEE MultiMedia*, 18(4):62–67.

Stockhammer, T. (2011). Dynamic adaptive streaming over http –: Standards and design principles. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 133–144, New York, NY, USA. ACM.

Wichtlhuber, M., Reinecke, R., and Hausheer, D. (2015). An sdn-based cdn/isp collaboration architecture for managing high-volume flows. *IEEE Transactions on Network and Service Management*, 12:15.

Uma Solução para Potencializar a Adaptação e Gerenciamento de Funcionalidades de Infraestruturas Baseadas em SDN

Emídio P. Neto[1], Felipe S. Dantas Silva[1,2], Marcilio O. O. Lemos[1,2]
Augusto Venâncio Neto[2,3]

[1]LaTARC Research Lab
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte (IFRN)
Natal-RN, Brasil

[2]Departamento de Informática e Matemática Aplicada (DIMAp)
Universidade Federal do Rio Grande do Norte (UFRN) – Natal-RN, Brasil

[3]Instituto de Telecomunicações (IT), Aveiro, Portugal

emidio.neto@ifrn.edu.br, felipe.dantas@ifrn.edu.br
marcilio.cc.lemos@gmail.com, augusto@dimap.ufrn.br

Abstract. *The lack of standardization of the SDN Controller APIs requires high-level customization of the applications built under the control plane for the APIs specific to each adopted controller. This results in poor utilization of network features and functionality as well as reworking for infrastructure operators. In this context, this work proposes ATOM, an architecture for managing SDN controllers that allows network management operations to be performed by an external application that enables the management of network resources dynamically and ubiquitously. The proposal was validated through implementations to manage the Ryu controller and the preliminary results, considering specific tasks of the creation of flows, revealed a reduction in processing time for related operations.*

Resumo. *A ausência de padronização das APIs dos controladores SDN exige um alto nível de customização das aplicações contruídas sob o plano de controle para as APIs específicas de cada controlador adotado. Isso resulta em mau aproveitamento de recursos e funcionalidades da rede bem como retrabalho aos operadores da infraestrutura. Neste contexto, este trabalho propõe o ATOM, uma arquitetura para gerenciamento de controladores SDN que permite que as operações de gerenciamento da rede possam ser realizadas por uma aplicação externa que viabiliza o gerenciamento dos recursos de rede de forma dinâmica e ubíqua. A proposta foi validada através de implementações para gerenciamento do controlador Ryu e os resultados preliminares, considerando tarefas específicas de criação de fluxos, relevaram uma redução no tempo de processamento para as operações relacionadas.*

1. Introdução

Embora o paradigma das Redes Definidas por Software (SDN, do inglês *Software-Defined Networking*) [Haleplidis et al. 2015] proporcione inúmeras facilidades e flexibilidade para adaptação de novos serviços e aplicações, vários aspectos relacionados

ao gerenciamento da infraestrutura permanecem como grandes desafios, principalmente tratando-se da integração de diferentes mecanismos [Sezer et al. 2013]. Isto aplica-se, particularmente, à falta de sistemas de controle que possam suportar os administradores de rede quando confrontados com a crescente complexidade da infraestrutura e com a vasta gama de requisitos de qualidade de experiência dos usuários [Sharma et al. 2013].

Outro ponto, normalmente relacionado ao gerenciamento de infraestruturas SDN, é a necessidade da visualização da topologia e consequentemente interação com os dispositivos. Neste caso, os operadores de rede que não têm muito conhecimento no desenvolvimento de ferramentas e no manuseio de determinadas Interfaces de Programação de Aplicação (APIs, do inglês *Application Program Interface*) providas pelos controladores SDN irão encontrar dificuldades para lidar com a alta granularidade de informações que constituem o ecossistema SDN. Como resultado desta alta complexidade do gerenciamento das informações, os controladores SDN atualmente disponíveis exigem que o operador da rede tenha um profundo conhecimento relacionado as suas APIs e principais métodos de extensão de funcionalidades. Tal exigência pode acarretar em diversos problemas, tais como o mau aproveitamento das funcionalidades de controle, aumento da complexidade de gerenciamento e menor flexibilidade para adaptação de novas aplicações. Esta limitação se dá pelo motivo que cada controlador é implementado de maneira distinta, sem a preocupação com questões de padronização e interoperabilidade.

Em se tratando da orquestração de serviços de rede, a obtenção de informações da infraestrutura em tempo real é um fator crítico para qualquer cenário [Guck et al. 2017]. No entanto, a grande maioria das APIs de serviços disponibilizados pelos controladores não dispõe informações de *status* da topologia em tempo real, exigindo do desenvolvedor a implementação de métodos que atuem como ouvintes de eventos (do inglês *event listener*), que irão receber informações em tempo real dos eventos da rede e disponibilizá-las através de uma determinada aplicação. Tais informações serão posteriormente utilizadas como fonte de dados em tempo real. A ausência de mecanismos capazes de lidar com tais limitações implica uma série de problemas referentes ao monitoramento de serviços de rede, restringindo o acompanhamento em tempo real de performance de rede, escalabilidade e utilização de recursos.

Mecanismos relacionados a visualização e interação com a topologia de rede são funcionalidades fundamentais de uma ferramenta de orquestração SDN, cujo propósito é auxiliar o gerenciamento da topologia e reduzir a complexidade das operações realizadas em diversos tipos de dispositivos [Gomes et al. 2016]. Desta forma, uma solução viável para sanar as limitações mencionadas anteriormente é a implementação de interfaces de gerenciamento que, através das *Northbound Interfaces* dos controladores SDN, permitem a interação entre o operador de rede e a topologia em execução, além de disponibilizar interfaces padronizadas para desenvolvimento de inovadoras funções de rede.

Por meio de um levantamento recente, conduzido para estudo do estado da arte referente a orquestração de infraestruturas SDN, foi identificado que ambos a comunidade científica e industrial têm se dedicado a desenvolver novas soluções concentradas no controlador em uso. Neste contexto, as ferramentas são orientadas em cada mecanismo de orquestração, o que confirma as limitações descritas anteriormente e, consequentemente, podem restringir a atuação destas soluções em cenários mais complexos.

Tais limitações são as principais motivações deste estudo, uma vez que o processo de orquestração de infraestruturas SDN requer um conjunto de funcionalidades integradas que possam fornecer o suporte adequado para as atividades de gerência da rede. Diante disso, este trabalho supre esta lacuna ao propor o ATOM (A*n* ***A****rchitecture for Ubiqui****TO****us* ***M****anagement of SDN Controllers*), uma arquitetura modular baseada em componentes que, por meio de interfaces de comunicação em alto nível, é capaz de integrar múltiplas funcionalidades de gerenciamento de rede, permitindo que operações não triviais sejam executadas de forma prática e em tempo de execução. Através de uma interface única de gerenciamento, o ATOM viabiliza a gerência de estado dos dispositivos da rede e disponibiliza mecanismos para gerir os recursos da rede em diversos controladores SDN de forma ubíqua.

Com o intuito inicial de fornecer uma prova de conceito (PoC, do inglês *Proof of Concept*) da proposta da arquitetura ATOM, as implementações concentraram-se no controlador Ryu[1], amplamente utilizado no meio acadêmico-científico. As validações foram conduzidas em simulações (por meio da virtualização das infraestruturas no emulador Mininet[2]) e experimentações através de um *testbed* SDN real. O PoC foi validado por meio da implementação de dois módulos de serviço principais do ATOM: *(i)* Controlador de Topologia (*Topology Manager*), que inclui funcionalidades de gerência de fluxos e recuperação de falhas (*failover*); e *(ii)* Controlador de QoS (*QoS Manager*), que inclui funcionalidades de criação de políticas de QoS e configuração de filas por classes de tráfego.

O restante deste trabalho está organizado da seguinte maneira: a seção 2 aponta os trabalhos relacionados ao tema de pesquisa descrito, de modo a delinear as contribuições desta proposta. A seção 3 apresenta a proposta da arquitetura ATOM, expondo seus principais componentes e funcionalidades. A seção 4 descreve em detalhes uma das principais operações do ATOM e apresenta resultados da avaliação no que diz respeito as operações de inserção de fluxos. Por fim, a seção 5 apresenta as considerações finais e delineia as perspectivas de trabalhos futuros.

2. Trabalhos Relacionados

A literatura revela que várias das soluções de orquestração de controladores SDN existentes têm o escopo voltado apenas para o provimento de meios de visualização gráfica dos dispositivos de rede. Em algumas propostas as ferramentas exibem a forma como estão organizados os dispositivos presentes na topologia mas não são capazes de permitir a interação do operador com os elementos de rede por meio da interface de gerenciamento, como é o caso das propostas apresentadas em [Huang et al. 2014] e [Salsano et al. 2015]. Em [Schultz et al. 2015] os autores propõem o OpenGUFI, um sistema que disponibiliza uma interface gráfica (GUI, do inglês *Graphical User Interface*) para abstração de topologia de rede: a ferramenta permite que o operador visualize as conexões entre clientes móveis e pontos de acesso sem fio, com suas respectivas regras de fluxo, mas sem viabilizar meios de interação ao operador. Desta maneira, a proposta não aborda a problemática relacionada a complexidade de gerenciamento da topologia, que ainda tem que ser operada manualmente pelo administrador.

[1]https://osrg.github.io/ryu/
[2]http://mininet.org/

Em outro grupo de trabalhos anteriores as propostas foram projetadas por meio da integração com ferramentas disponibilizadas por terceiros. Essas soluções, na maioria das vezes, estão disponíveis através de uma licença de uso restrito, como é o caso do trabalho proposto em [Shalimov et al. 2015], no qual os autores propõem o controlador RUNOS. Tal proposta faz uso dos mecanismos disponibilizados pelo software de álgebra computacional *Maple* [Voellmy et al. 2013] para prover otimização às aplicações e adota o conceito de *multi-threading* efetivo para gerenciamento de processos como meio de evitar sobrecarga do sistema. No entanto, ao exigir que o operador de rede (que deseje fazer uso da solução) tenha um conhecimento especializado dos métodos restritos a determinadas APIs do próprio controlador, ele limita seu escopo no que se refere ao gerenciamento flexível e ubíquo.

Outra abordagem consiste no desenvolvimento de soluções de orquestração integradas em controladores específicos, que precisam ser adaptados ao contexto da solução de orquestração. Isso está sujeito a restrições graves de gerenciamento de rede, uma vez que aplicações que foram desenvolvidas para um controlador específico só podem ser usadas por ele, impedindo a interoperabilidade entre os controladores. Algumas soluções baseadas nesta perspectiva são descritas abaixo.

A proposta em [Koponen et al. 2010] apresenta o sistema Onix, que propõe a estruturação em *cluster* de um ou mais controladores NOX [Gude et al. 2008]. Tal abordagem, faz uso de uma Base de Informações de Rede (NIB, do inglês *Network Information Base*) em conjunto com o Apache ZooKeeper [The Apache Software Foundation 2016] e utiliza um protocolo proprietário para comunicação horizontal entre os controladores que formam o *cluster*. Além disso, a implementação principal não está disponível como um software de código aberto, o que certamente desencoraja a integração e o desenvolvimento de novos serviços e aplicações baseados nesse sistema.

Em [Choi et al. 2015] os autores apresentam o IRIS-CoMan como uma solução para o gerenciamento de infraestrutura SDN em larga-escala. A proposta apresentada é baseada no controlador Floodlight e requer a migração de todas as aplicações já desenvolvidas para manter conformidade com a API disponibilizada.

Os autores em [Hassas Yeganeh and Ganjali 2012] introduzem o Kandoo, que propõe uma estruturação para os controladores baseada em uma arquitetura hierárquica e tem como objetivo fornecer um controlador principal na parte superior da rede. Desta maneira espera-se viabilizar a interconexão de vários outros controladores. Esta abordagem é restrita ao controlador Kandoo e faz uso de um protocolo proprietário para comunicação entre as entidades controladoras, sendo inadequada para infraestruturas que fazem uso de outros controladores SDN.

DISCO [Phemius et al. 2013] propõe compartilhar o *status* das conexões de rede, mantendo a comunicação entre todos os controladores vizinhos. Além disso, esta abordagem requer que o código fonte das aplicações sejam atualizados no controlador Floodlight, o que torna a proposta completamente dependente deste controlador.

O HyperFlow [Ganjali and Tootoonchian 2010] está disponível como uma aplicação do controlador NOX [Gude et al. 2008] e requer a atualização de várias linhas de código no controlador, de modo que seja possível interceptar comandos e serializar

eventos. Por esta razão, esta abordagem revelou-se como sendo uma solução que está disponível apenas para sanar as adversidades de um controlador específico (o NOX).

Os autores em [Frate et al. 2018] propuseram uma ferramenta para gerenciamento de controladores OpenFlow que utiliza-se das REST APIs dos controladores Ryu e Floodlight para obtenção de informações de eventos da rede. Para tal, faz uso de um banco de dados orientado a grafos (Neo4J[3]) para armazenamento dos dados da topologia. Em consequência, a proposta torna-se totalmente dependente das funcionalidades providas pelo Neo4J para disponibilização da interface de gerenciamento, o que dificulta a implementação de recursos adicionais que possam ser providos pela interface de gerenciamento.

Embora as propostas referenciadas nesta seção apresentem seus respectivos benefícios, elas não se preocupam com a possível convergência entre diversos tipos de controladores SDN atualmente disponíveis. Algumas das propostas disponibilizam apenas funcionalidades de visualização gráfica da topologia em execução e são restritas a um controlador específico, não apresentando em suas arquiteturas métodos ou APIs que possibilitem escalabilidade relacionada ao número de controladores gerenciados. Outro grupo de propostas apresentam dependência de softwares que necessitam de licença de uso e dependência de protocolos proprietários, desencorajando seu uso em ambientes de larga escala.

A análise da literatura relacionada revelou que as soluções propostas não fornecem funcionalidades mínimas de orquestração ao operador de rede no que diz respeito ao gerenciamento ubíquo da infraestrutura. Tarefas fundamentais como gerenciamento de fluxos e de políticas de QoS devem ser disponibilizadas no mais alto nível de abstração. Diante disso, espera-se que as próximas gerações de soluções de orquestração sejam capazes de prover, no mínimo: *(i)* visualização em tempo real da topologia de rede; *(ii)* mecanismos de interação com a topologia em execução; e *(iii)* independência de softwares proprietários. A Tabela 1 apresenta uma comparação dos trabalhos anteriores discutidos nesta seção em relação com a proposta apresentada neste trabalho (sistema ATOM), levando em consideração os aspectos básicos de funcionalidades mínimas esperadas de uma solução de orquestração de controladores SDN.

A relação abaixo apresenta uma descrição de cada item utilizado como parâmetro de comparação:

- **T1 – Visualização de Topologia**: prover mecanismo de visualização de topologia para o operador de rede;
- **T2 – Interação com a Topologia**: prover, com interatividade, mecanismos básicos de gerenciamento de topologia e de QoS;
- **T3 – Independência de Softwares Proprietário**: independência de mecanismos de terceiros, de controladores SDN e protocolos proprietários.

Conforme a Tabela 1, que agrupa o estado da arte atualmente disponível na literatura, é possível constatar que apenas 01 (OrchFlow [Frate et al. 2018]) dos 10 estudos realizados em trabalhos anteriores apresentou evidências de ter se comprometido na concepção de um sistema para gerenciamento de controladores SDN. Todavia, a proposta OrchFlow não menciona a possibilidade da interoperabilidade entre os controladores da

[3]https://neo4j.com/

rede para fornecer recursos como meio de possibilitar a adaptação de funcionalidades inovadoras por meio de interfaces em alto nível.

Tabela 1. Comparativo entre os trabalhos relacionados e a proposta do sistema ATOM

Proposta	T1	T2	T3
HyperFlow, [Ganjali and Tootoonchian 2010]	✓	X	X
Onix, [Koponen et al. 2010]	✓	✓	X
Kandoo, [Hassas Yeganeh and Ganjali 2012]	✓	X	X
DISCO, [Phemius et al. 2013]	✓	✓	X
[Huang et al. 2014] Mantoo, [Salsano et al. 2015] OpenGUFI, [Schultz et al. 2015]	✓	X	X
IRIS-CoMan, [Choi et al. 2015]	✓	X	X
RUNOS, [Shalimov et al. 2015]	✓	X	X
OrchFlow, [Frate et al. 2018]	✓	✓	X
ATOM	✓	✓	✓

A próxima seção introduz o sistema ATOM, cuja proposta baseada em uma arquitetura modular que viabiliza uma melhor organização dos componentes, se apresenta como uma alternativa promissora para lidar com as lacunas não preenchidas pela literatura relacionada.

3. Proposta do Sistema ATOM

Esta seção apresenta o sistema ATOM (An **A**rchitecture for Ubiqui**TO**us **M**anagement of SDN Controllers). ATOM atua no gerenciamento de controladores SDN e é capaz de prover interfaces de comunicação padronizadas e abstrair a complexidade de implementação e utilização das funcionalidades dos controladores. As seções seguintes fornecem uma visão geral da arquitetura e apresentam em detalhes as funcionalidades de cada componente.

3.1. Visão Geral

O sistema ATOM tem como principal objetivo abstrair a dificuldade encontrada atualmente para gerenciamento das funções fornecidas pelos controladores SDN e, em paralelo, fornecer ao operador da rede uma visão em alto nível de toda a infraestrutura que está sob o domínio do controlador. Para isso, é necessário o provisionamento de interfaces de comunicação e módulos que proporcionem interoperabilidade entre o controlador e os componentes da ferramenta. A fim de prover tais funcionalidades, faz-se necessário a adição de uma nova camada que agrega os componentes responsáveis pelo gerenciamento dos recursos fornecidos pelos controladores SDN. A Figura 1 apresenta a arquitetura do sistema ATOM posicionada sob uma típica infraestrutura SDN, composta pelo controlador, demais elementos do plano de dados, interfaces de comunicação e protocolos utilizados.

ATOM é formado por um conjunto de componentes que convergem a fim de obter e agregar as informações de toda a topologia, incluindo os eventos de rede. Estas

informações são obtidas dos controladores SDN e, após abstração da complexidade relacionada à leitura e processamento, são disponibilizadas para as aplicações. Numa visão de alto nível, os componentes da arquitetura são descritos da seguinte maneira: *(i) Interface Management*; *(ii) Resource Manager*; *(iii) Server Provider*; *(iiii) Broker Server Interface*; e *(iv) Broker Client Interface*.

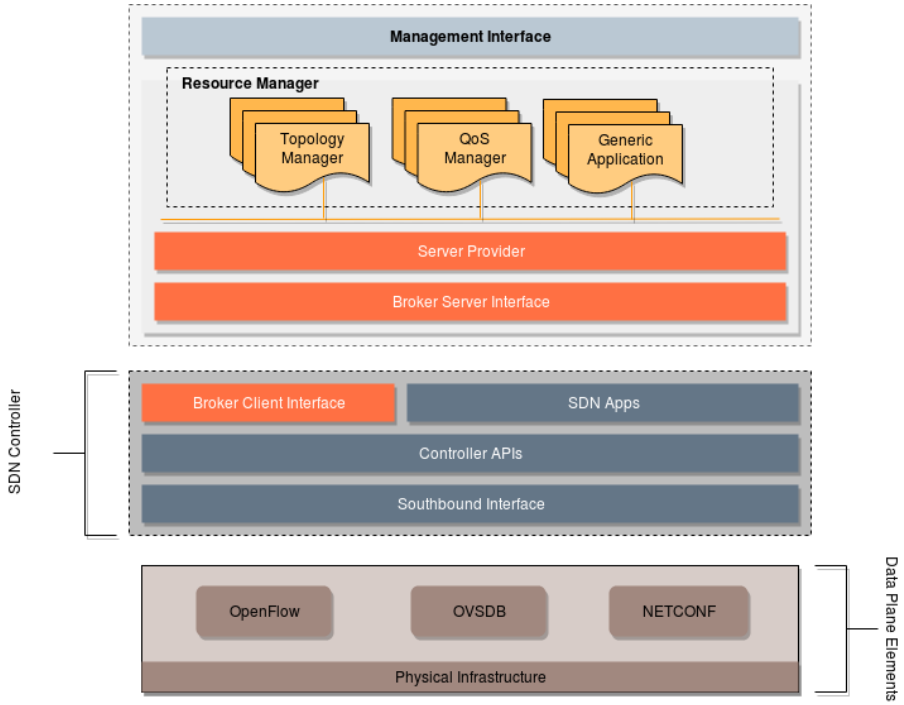


Figura 1. Arquitetura do sistema ATOM

3.1.1. Resource Manager

O *Resource Manager* é o módulo responsável por agregar os componentes de *Topology Manager* e *Qos Manager* para fornecer uma interface de comunicação com o intuito de disponibilizar as funcionalidades providas por tais componentes.

3.1.2. Topology Manager

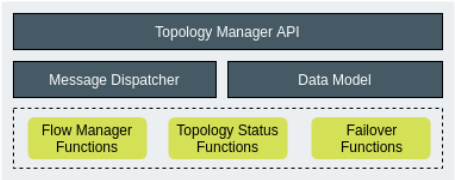


Figura 2. Arquitetura interna do *Topology Manager*

Componente responsável por três principais tarefas: *(i)* obtenção do *status* da topologia de rede em tempo real; *(ii)* provimento de funções básicas para gestão da rede

(ex. gerenciamento de fluxos); *(iii)* provimento de mecanismos para detecção de falha de comunicação entre os enlaces (*failover*). De acordo com a Figura 2, o *Topology Manager* é estrutura da seguinte maneira:

- ***Topology Manager REST API***: interface de comunicação utilizada pela interface de gerenciamento em alto nível (Management Interface);
- ***Message Dispatcher***: funções responsáveis pelo encaminhamento das requisições geradas pelas operações de gerenciamento para o Server Provider;
- ***Data Model***: modelo responsável pela definição da estruturação das requisições que são recebidas pelo *Topology Manager* e enviadas para o *Server Provider*;
- ***Flow Manager Functions***: disponibiliza as funções para gerenciamento das regras de fluxo;
- ***Topology Status Functions***: disponibiliza as funções para obtenção e gerenciamento do estado atual da topologia;
- ***Failover Functions***: funções responsáveis pela identificação (e aplicação de contramedidas) de falha em enlaces de comunicação.

3.1.3. QoS Manager

O *QoS Manager* viabiliza a criação e manutenção de políticas de QoS, possibilitando ao operador de rede a opção de configuração de reservas de recursos em caminhos selecionados. Por meio de uma abordagem simplificada, que modela os dados necessários para a operação desta funcionalidade, o operador de rede deverá apenas informar as especificações de QoS desejadas, conforme a Tabela 2.

Tabela 2. Especificação das requisições QoS Manager

Campo	Tipo	Descrição
portName	text	Porta do dispositivo a qual a política será aplicada.
type	linux-htb	Escalonador de filas.
maxRate	bits	Limite máximo de *bits* aplicados na política.
minRate	*bits*	Limite mínimo de bits aplicados na política.

3.1.4. Server Provider

A fim de disponibilizar todas as funcionalidades e informações coletadas pelos componentes previamente descritos, um sistema de *backend* apropriado é necessário para realizar todo o processamento de informações. Desta forma, o *Server Provider*, que é o ponto principal de comunicação entre o controlador SDN e os demais elementos da arquitetura, é o componente responsável por suportar toda a estruturação de componentes do sistema ATOM.

O *Server Provider* é capaz de padronizar as requisições do *Topology Manager* e do *QoS Manager*. O processo de padronização é conduzido através de um mapeamento, que converte as requisições para uma linguagem de texto com sintaxe de fácil leitura baseada no formato *JavaScript Object Notation* (JSON). Sendo assim, quando algum componente

necessita se comunicar com o controlador SDN, o *Server Provider* executa o procedimento de estruturação das mensagens. Após isso, realiza a verificação de consistência de tais informações e, em caso positivo, encaminha a requisição para o controlador SDN.

Neste contexto, o *Server Provider* pode ser visto como um intermediador entre todas as informações trafegadas entre o ATOM e o controlador SDN, uma vez que não necessita tomar conhecimento de qual o tipo de controlador ele está recebendo e solicitando informações. Por realizar o tratamento de requisições de forma assíncrona, ele é capaz de atender a requisições de controladores de forma indistinta.

3.1.5. *Broker Interfaces*

Para cumprir o objetivo da proposta, é necessário a implementação de uma camada de adaptadores para prover a intercomunicação entre o *Server Provider* e o controlador SDN. O componente responsável por tal adaptação recebe o nome de *Broker*. A arquitetura do sistema ATOM é composta por dois tipos de *Broker*:

1. ***Broker Server*** – responsável por receber as informações padronizadas pelo *Server Provider* e enviá-las para o controlador SDN.
2. ***Broker Client*** – responsável por executar, no lado do controlador SDN, as instruções recebidas pelo *Broker Server*.

3.1.6. Management Interface

Este componente é definido por uma GUI, projetada para prover ao operador de rede a visão holística de toda a infraestrutura que está sob o domínio administrativo de um determinado controlador. Através do *Management Interface* o operador da rede não necessitará realizar intervenções manuais por meio de operações em interfaces de linhas de comando (CLI, do inglês *Command-Line Interface*) dos controladores ou dos elementos de rede. Por meio desta abordagem as funcionalidades providas pelo *Resource Manager* podem ser aplicadas diretamente aos dispositivos de rede.

O *Management Interface* também é capaz de abstrair a complexidade de leitura de informações providas pelos controladores SDN, exigindo pouca intervenção manual para que seja possível a visualização de informações, a exemplo: *(i)* regras de fluxos ativas nos dispositivos de rede; *(ii)* estatísticas de tráfego dos enlaces; e *(iii)* políticas de QoS.

4. Operações e Avaliação

ATOM fornece ao operador de rede a possibilidade de gerenciamento de fluxos customizados (criação e remoção) para atender demanda específicas das aplicações. A Figura 3 detalha a sequência de operações e mensagens trocadas entre os componentes envolvidos no procedimento de criação de fluxos.

Conforme descrito na Figura 3, o operador de rede, no primeiro momento, requisita a instalação de um novo fluxo com parâmetros customizados. A requisição é então enviada para o *Topology Manager* (passo 2.1) e, por meio de seus componentes internos (*Topology Manager API* e *Message Dispatcher*), realiza a tradução e o encaminhamento da mensagem para o componente responsável pela estruturação da requisição (*Topology*

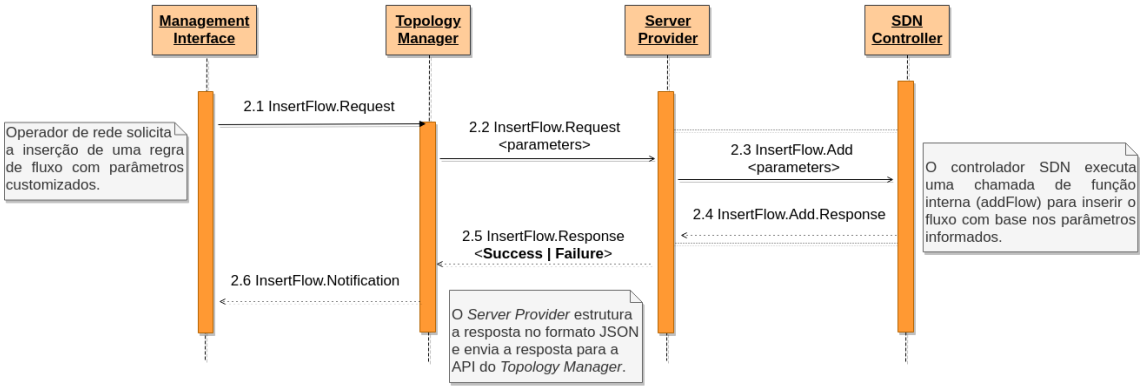


Figura 3. Diagrama de sequência do procedimento de inserção de fluxos

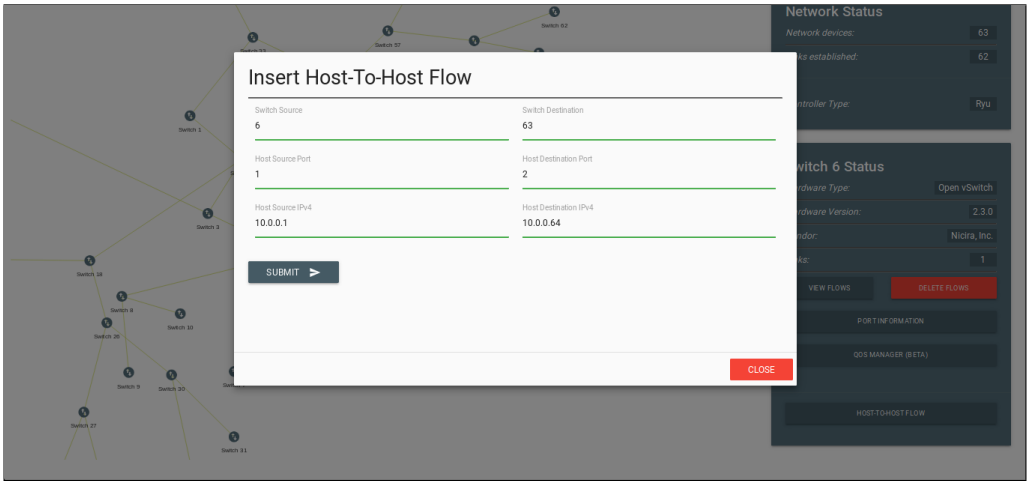


Figura 4. Visualização da interface de inserção de fluxos

Status Functions). Em seguida, a requisição é enviada para o controlador SDN, por meio do*Server Provider* (passo 2.2). O controlador SDN traduz a requisição em mensagens `FlowMod`) e executa uma chamada interna para adicionar uma nova entrada na tabela de fluxos (passo 2.3) do dispositivo indicado durante a solicitação de criação do fluxo. Após isso, o controlador envia uma mensagem ao *Server Provider* (passo 2.4) contendo o resultado do procedimento (*Success* ou *Failure*). Em ambos os casos, o operador sera notificado por meio do *Management Interface* (passo 2.6). As Figuras 4 e 5 demonstram as funcionalidades providas pelo *Management Interface* para os procedimentos de inserção e visualização das tabelas de fluxo.

Uma série de experimentos foram conduzidos com o objetivo de validar a eficiência do sistema ATOM no que diz respeito ao tempo necessário para realizar os procedimentos de orquestração por meio do *Management Interface*. Foram utilizadas 04 infraestruturas de rede com topologias configuradas de forma diferente, conforme a Tabela 3.

As infraestruturas foram logicamente virtualizadas através do emulador Mininet[4]. As avaliações foram conduzidas em um *host laptop* Dell Inspiron, Intel® Core™ i7-

[4]http://mininet.org

Tabela 3. Configuração dos cenários de avaliação

Topologia	Switches	Nº de Saltos	Nº de Fluxos**
T1	15	14	30
T2	25	24	50
T3	35	34	70
T4	45	44	90

** Representa o número de fluxos a serem instalados no experimento.

5500U, 8GB RAM, 500GB de disco rígido e sistema operacional Linux Debian 8.

Os experimentos levaram em consideração o procedimento de inserção de fluxos fim a fim entre *hosts* localizados em extremidades opostas da rede. Para isso, todas as topologias descritas na Tabela 3 foram utilizadas, e o mesmo processo foi executado 20 vezes para cada uma delas. Tomando como exemplo a topologia **T1**, que é composta por 15 switches e 2 *hosts* (separados por 14 saltos de distância), 30 regras de fluxos são necessárias para prover a comunicação entre os *hosts* (fluxos de ida e fluxos de volta). A Figura 6 revela os resultados obtidos para cada topologia a partir dos tempos médio gastos para a instalação das respectivas regras de fluxos.

De acordo com os resultados mostrados na Figura 6, a inserção de fluxos provida pelas funcionalidades do sistema ATOM foi realizada em um tempo menor do que na abordagem tradicional (fazendo uso de scripts baseados em Python [5]). A Tabela 4 apresenta um comparativo entre os resultados obtidos (a partir da média aritmética simples dos resultados obtidos em cada uma das 20 execuções para cada topologia) na avaliação e o percentual de otimização alcançado pela abordagem utilizada no sistema ATOM.

Embora a organização da arquitetura do sistema ATOM sugira a perspectiva de maior latência nas atividades de orquestração, os resultados das avaliações mostraram que os benefícios vão além das funcionalidades descritas nas seções anteriores. No caso deste experimento, ao executar apenas uma requisição que insere todos os fluxos em um

[5]Scripts disponível em: https://github.com/latarc/expsltrc

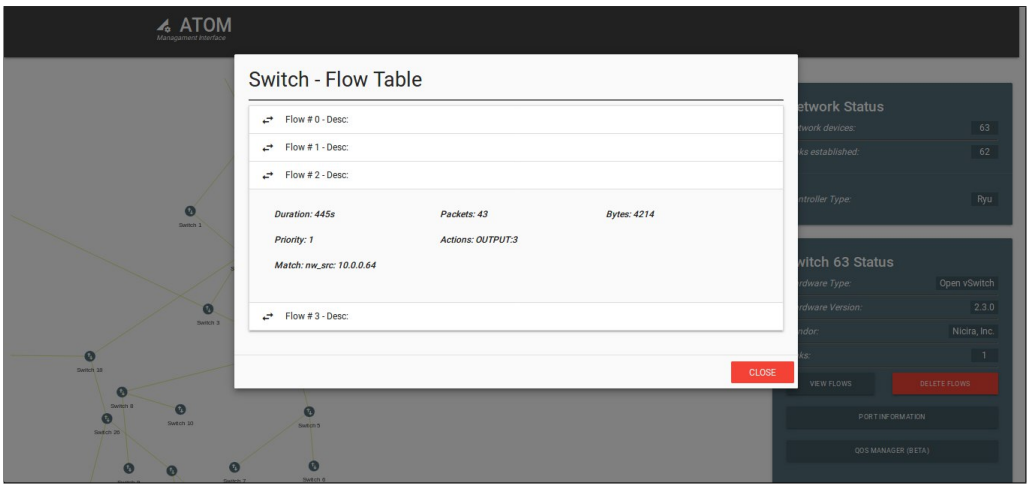


Figura 5. Visualização de informações das tabelas de fluxo dos dispositivos

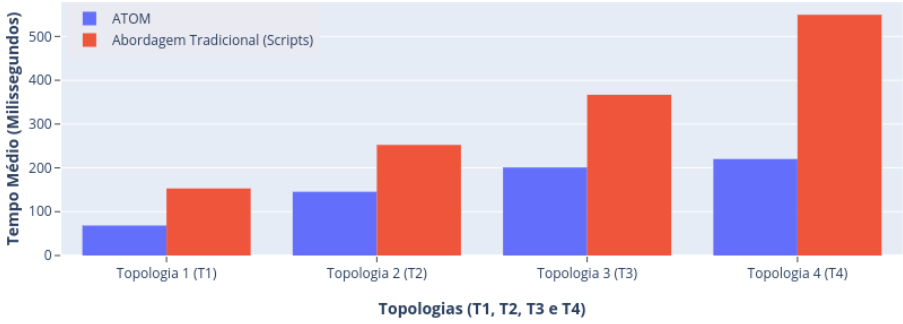


Figura 6. Experimento para avaliação da inserção de fluxos

Tabela 4. Resultados do experimento de inserção de fluxos

Topologia	ATOM	Inserção manual	Otimização provida por ATOM
T1	68.4ms	153.0ms	55.3%
T2	145.4ms	252.4ms	42.4%
T3	201.0ms	366.8ms	45.21%
T4	220.2ms	549.4ms	59.92%

determinado caminho (ATOM), o tempo de processamento das mensagens e da inserção de fluxos acaba sendo inferior ao tempo gasto pela inserção manual devido ao processamento único da requisição (necessário para prosseguir com a configuração dos fluxos em um determinado caminho). Em contraste, a inserção de fluxos manuais necessitará de um requisição para cada fluxo a ser instalado, o que pode representar um problema de escalabilidade devido ao número de dispositivos existentes na topologia.

5. Considerações Finais

Neste trabalho apresentamos o ATOM, uma arquitetura modular capaz de maximizar a capacidade de orquestração de controladores SDN, permitindo uma melhor utilização dos recursos de rede e na redução de execução de operações manuais. Por meio da separação das funcionalidades em componentes especializados, ATOM é capaz de prover a interação com os dispositivos da rede de forma direta. Através das especificações da arquitetura é possível disponibilizar uma interface de gerenciamento para viabilizar o gerenciamento holístico da infraestrutura que está associada a um determinado controlador.

ATOM foi projetado para implementar, de forma padronizada, o fluxo de mensagens relacionados às requisições necessárias para configuração das aplicações. Desta forma, é possível implementar módulos que utilizam as *Northbound Interfaces* disponibilizadas pelos controladores SDN para proporcionar o gerenciamento da topologia em tempo real.

Para os trabalhos futuros pretende-se aprimorar o gerenciamento das interações entre o ATOM e os controladores a fim de proporcionar melhor experiência para adaptação de novas funcionalidades. Dentre as principais, destaca-se a otimização dos

mecanismos disponibilizados pelo *Topology Manager*, em específico para o mecanismo de *failover*, para realizar a seleção do caminho ótimo baseado na disponibilidade de recursos dos enlaces. Além disso, tenciona-se estender o conceito de gerenciamento ubíquo provido por ATOM para o gerenciamento agnóstico de controladores. Desta maneira, esperamos fornecer a capacidade de orquestração de múltiplos controladores diferentes a partir de uma mesma *Management Interface* através da adoção de templates previamente configurados que irão definir a forma de comunicação com os controladores SDN, disponibilizando interfaces agnósticas para tal comunicação. Futuras avaliações consistirão de analises das demais operações fornecidas pelo ATOM, a fim de mensurar o tempo de gerenciamento em diferentes cenários para servir como prova de conceito para utilização do trabalho proposto.

6. Agradecimentos

Esta pesquisa foi parcialmente apoiada pela 4ª chamada colaborativa EU-BR (H2020, *grant agreement* no. 777067, NECOS – *Novel Enablers for Cloud Slicing*), financiada pela Comissão Européia e pelo Ministério da Ciência, Tecnologia, Inovação e Comunicação (MCTIC), através da RNP e CTIC. Os autores também agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e ao IFRN pelo apoio.

Referências

Choi, T., Lee, B., Kang, S., Song, S., Park, H., Yoon, S., and Yang, S. (2015). Iris-coman: Scalable and reliable control and management architecture for sdn-enabled large-scale networks. *J. Netw. Syst. Manage.*, 23(2):252–279.

Frate, M., Marczuk, M. K., and Verdi, F. L. (2018). OrchFlow: An Architecture for Orchestration of Multiple Controllers in OpenFlow Networks. *Journal of Network and Systems Management*, (0123456789).

Ganjali, Y. and Tootoonchian, A. (2010). Hyperflow: A distributed control plane for openflow. In Akella, A., Feamster, N., and Rao, S. G., editors, *INM/WREN*. USENIX Association.

Gomes, C. S., Silva, F. S. D., Neto, E. P., Costa, K. B., and da Silva, J. B. (2016). Towards a modular interactive management approach for sdn infrastructure orchestration. In *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 1–6.

Guck, J. W., Bemten, A. V., and Kellerer, W. (2017). Detserv: Network models for real-time qos provisioning in sdn-based industrial environments. *IEEE Transactions on Network and Service Management*, 14(4):1003–1017.

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., and Shenker, S. (2008). Nox: Towards an operating system for networks. *SIGCOMM Comput. Commun. Rev.*, 38(3):105–110.

Haleplidis, E., Pentikousis, K., Denazis, S., Salim, J. H., Meyer, D., and Koufopavlou, O. (2015). Software-Defined Networking (SDN): Layers and Architecture Terminology. RFC 7426.

Hassas Yeganeh, S. and Ganjali, Y. (2012). Kandoo: A framework for efficient and scalable offloading of control applications. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, pages 19–24, New York, NY, USA. ACM.

Huang, W.-Y., Chou, T.-Y., Hu, J.-W., and Liu, T.-L. (2014). Automatical end to end topology discovery and flow viewer on sdn. In *Proceedings of the 2014 28th International Conference on Advanced Information Networking and Applications Workshops*, WAINA '14, pages 910–915, Washington, DC, USA. IEEE Computer Society.

Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., Ramanathan, R., Iwata, Y., Inoue, H., Hama, T., and Shenker, S. (2010). Onix: a distributed control platform for large-scale production networks. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI'10, pages 1–6, Berkeley, CA, USA. USENIX Association.

Phemius, K., Bouet, M., and Leguay, J. (2013). Disco: Distributed multi-domain sdn controllers. *CoRR*, abs/1308.6138.

Salsano, S., Ventre, P. L., Lombardo, F., Siracusano, G., Gerola, M., Salvadori, E., Santuari, M., Campanella, M., and Prete, L. (2015). Mantoo - a set of management tools for controlling sdn experiments. In *Proceedings of the 2015 Fourth European Workshop on Software Defined Networks*, EWSDN '15, pages 123–124, Washington, DC, USA. IEEE Computer Society.

Schultz, J., Szczepanski, R., Haensge, K., Maruschke, M., Bayer, N., and Einsiedler, H. (2015). Opengufi: An extensible graphical user flow interface for an sdn-enabled wireless testbed. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, pages 770–776.

Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., Viljoen, N., Miller, M., and Rao, N. (2013). Are we ready for sdn? implementation challenges for software-defined networks. *IEEE Communications Magazine*, 51(7):36–43.

Shalimov, A., Nizovtsev, S., Morkovnik, D., and Smeliansky, R. L. (2015). The runos openflow controller. In *EWSDN*, pages 103–104. IEEE Computer Society.

Sharma, P., Banerjee, S., Tandel, S., Aguiar, R., Amorim, R., and Pinheiro, D. (2013). Enhancing network management frameworks with sdn-like control. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 688–691.

The Apache Software Foundation (2016). Apache ZooKeeper. [Online] Accessed 4 Mar 2016. Available at: `https://zookeeper.apache.org/`.

Voellmy, A., Wang, J., Yang, Y. R., Ford, B., and Hudak, P. (2013). Maple: Simplifying sdn programming using algorithmic policies. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 87–98, New York, NY, USA. ACM.

Uma Solução Flexível e Personalizável para a Composição de Cadeias de Função de Serviço

Vinícius F. Garcia[1], Marcelo Caggiani Luizelli[2], Elias P. Duarte Júnior [1], Carlos Raniery P. dos Santos[3]

[1]Universidade Federal do Paraná (UFPR)
Caixa Postal 19018 81531-990, Curitiba - PR

[2]Universidade Federal do Pampa (UNIPAMPA)
Av. Tiaraju, 810 - Ibirapuitã, Alegrete - RS

[3]Universidade Federal de Santa Maria (UFSM)
Av. Roraima, 1000 - Camobi, Santa Maria - RS

vfgarcia@inf.ufpr.br, marceloluizelli@unipampa.edu.br,
elias@inf.ufpr.br, csantos@inf.ufsm.br

Abstract. *One of the greatest challenges of the Network Function Virtualization paradigm is related to virtualized services deployment (composition, embedding, and scheduling). In particular, the service function chain composition process is currently executed by solutions that are static and do not present customizable operational behavior and evaluation metrics. This paper proposes a new service composing solution that is able to deal with multiple requirements from network operators. An adaptative methodology is employed in order to evaluate an elastic and heterogeneous metric set, conciliating different granularities and semantics. Finally, results obtained on an experimental environment show the feasibility of the proposed solution for specific services composition.*

Resumo. *Um dos principais desafios do paradigma de Virtualização de Funções de Rede reside na implantação de serviços virtualizados (composição, integração e execução). Em especial, a composição de cadeias de função de serviço é atualmente realizada por soluções estáticas e não personalizáveis em relação aos seus comportamentos operacionais e métricas de avaliação. Este artigo propõe uma nova solução de composição capaz de tratar requisitos específicos dos operadores de redes. Para isso, uma metodologia adaptativa é adotada para avaliar um conjunto elástico de métricas heterogêneas, conciliando diferentes granularidades e semânticas. Finalmente, resultados obtidos em um ambiente experimental demonstraram a viabilidade da solução proposta na composição de serviços específicos.*

1. Introdução

As infraestruturas de telecomunicações dependem de diferentes Funções de Rede (*Network Functions* - NF) para operarem. Atualmente, essas NFs são majoritariamente implementadas em equipamentos dedicados – chamados de *physical appliances* – capazes de identificar, tratar e executar protocolos e serviços. Porém, tais equipamentos tipicamente apresentam alto custo financeiro, baixa escalabilidade e processos de

manutenção complexos, contribuindo com o problema conhecido como ossificação da Internet [Handley 2006]. Nesse contexto, o paradigma de Virtualização de Função de Rede (*Network Function Virtualization* - NFV) surge objetivando, através do uso de técnicas de virtualização já existentes, instanciar funções de rede sobre um plano de *software* (*Virtualized Network Functions* - VNF) em equipamentos de propósito geral [NFVISG 2012]. Dessa forma, são esperados como benefícios a redução de custos financeiros e a flexibilização do monitoramento e da manutenção de NFs, além da simplificação do processo de desenvolvimento e implantação de novos serviços de rede criados a partir de uma ou mais funções virtualizadas.

O paradigma NFV também prevê a criação de serviços de rede através da conexão de diferentes funções em uma estrutura de encaminhamento e processamento de tráfego conhecida como Cadeia de Funções de Serviço (*Service Function Chain* - SFC) [Quinn and Nadeau 2015]. A implantação de um serviço de rede em um ambiente virtualizado acontece a partir de uma série de etapas interrelacionadas que constituem o processo de implantação. Resumidamente, a implantação em NFV é chamada de Alocação de Recursos (NFV *Resource Allocation* - NFV-RA) [Herrera and Botero 2016] e é divida em três etapas: composição (*composition*), integração (*embedding*) e agendamento de execução (*scheduling*). Assim, os desafios particulares associados a cada etapa do NFV-RA vêm sendo extensivamente explorados pela comunidade acadêmica, resultando em soluções que são empregadas no cumprimento de cada uma delas [Gil-Herrera and Botero 2017, Ocampo et al. 2017, Baek et al. 2017, Allybokus et al. 2018, Faraci et al. 2016, Kulkarni et al. 2017].

Porém, em geral, a implantação é realizada considerando apenas a topologia da rede, os recursos físicos disponíveis e algumas outras características predefinidas. Assim, aspectos das próprias cadeias de serviço (*e.g.*, heterogeneidade de funções de rede, simetria de conexões, compartilhamento de funções virtualizadas) e possíveis necessidades específicas a serem definidas pelos clientes e/ou operadores de rede são desconsideradas. Em especial, soluções de composição existentes [Mehraghdam et al. 2014, Dräxler and Karl 2017, Gil-Herrera and Botero 2017, Ocampo et al. 2017, Wang et al. 2017] apresentam capacidade de análise limitada já que operam sobre um conjunto de métricas de avaliação rigidamente programado e imutável. Entre as métricas comumente observadas nessas soluções estão a taxa de tráfego, requisitos de banda e recursos computacionais, tamanho da cadeia de serviço e níveis de prioridade definidos para funções de rede. Apesar de formarem um conjunto significativo, limitar as soluções a essas métricas leva a uma execução inflexível, restritiva e sem suporte a personalização da etapa de composição. Dessa forma, o operador de rede é forçado a adequar suas necessidades às possibilidades de análise das soluções de composição existentes ou, para que o contrário ocorra, adaptar as mesmas em código.

Cenários de implantação são sobretudo formados pelos diferentes objetivos e necessidades dos operadores de rede (*e.g.*, redução da taxa de tráfego no serviço, do atraso de processamento, do consumo de memória, da quantidade de VNFs), por ambientes de virtualização heterogêneos (*e.g.*, centralizado, distribuído, privado, público) e por tipos variados de serviços (*e.g.*, roteamento, segurança, balanceamento de carga). Considerando a baixa capacidade adaptativa a esses cenários por parte das soluções de composição existentes, este trabalho apresenta o *CUstomizable Service COmposing* (CUSCO), uma

nova solução para a composição de serviços de rede virtualizados que aplica uma avaliação personalizada de métricas de interesse informadas pelo operador de rede. A solução CUSCO executa uma metodologia dinâmica e flexível de avaliação que possibilita a conciliação de múltiplas métricas com diferentes granularidades e semânticas, estas fornecidas através de um modelo de requisição próprio.

O restante deste artigo está estruturado como se segue: a Seção 2 examina e discute os principais trabalhos relacionados; a Seção 3 apresenta a nova proposta do CUSCO; a Seção 4 exibe os testes realizados; e, finalmente, a Seção 5 conclui o trabalho.

2. Trabalhos Relacionados

Nesta seção são apresentados os principais trabalhos que abordam a etapa de composição de cadeias de serviço. Uma vez que diferentes topologias podem ser construídas para prover um mesmo serviço de rede através de um grupo comum de funções virtualizadas [Herrera and Botero 2016], soluções de composição capazes de determinar, analisar e qualificar as possíveis topologias se tornam necessárias. Dessa forma, as soluções de composição recebem como entrada documentos de requisição com informações necessárias à sua execução (*e.g.*, funções de rede que compõem o serviço, seus comportamentos operacionais e suas inter-relações). Ainda, recomendações de implantação (*i.e.*, dados resultantes de uma investigação prévia que sugerem as configurações ideais para implantação do serviço em diferentes ambientes de execução) e informações obtidas através de *benchmarkings* das funções de rede virtualizadas [Rosa et al. 2015] são frequentemente utilizadas como objetos de análise na etapa de composição.

Em [Mehraghdam et al. 2014] apresenta-se uma solução de composição que requer uma cadeia de serviço parcialmente ordenada, além da taxa de tráfego das funções de rede que a compõem (*i.e.*, o quociente entre a quantidade de dados recebidos e quantidade de dados encaminhados). Nela, uma função objetivo de minimização da taxa de tráfego agregada é adotada através de uma heurística de adiantamento de possíveis descartes de pacotes. Da mesma forma, a solução descrita em [Dräxler and Karl 2017] também opera sobre uma topologia parcialmente ordenada. Entretanto, esta utiliza-se de um modelo de Pareto e simulações sucessivas de integração objetivando, assim, determinar a composição capaz de otimizar a relação custo/benefício entre três métricas que formam sua função objetivo (tamanho da cadeia, taxa de tráfego agregada e necessidade de recursos computacionais).

As soluções presentes em [Ocampo et al. 2017] e [Gil-Herrera and Botero 2017] avaliam os possíveis relacionamentos entre as funções de rede destinadas a executar um serviço. A primeira apresenta uma função objetivo, modelada por Programação Linear Inteira, para minimizar a taxa de tráfego agregada e a quantidade de recursos computacionais necessários para a integração do serviço requisitado. Já a segunda utiliza uma função objetivo que busca assegurar que seja mínima a necessidade de largura de banda do serviço requisitado, calculando a mesma através de uma meta-heurística baseada em busca tabu. Por fim, em [Wang et al. 2017] uma solução de composição automática é descrita. Essa solução determina as dependências de um serviço baseada nas características operacionais de suas funções de rede, visando assim a criação de topologias válidas. Para tanto, diagramas Hasse são usados na definição de ordenamentos e conexões considerando uma função objetivo de prioridade de VNFs informadas pelo operador de rede.

A Tabela 1 sumariza as principais características dos trabalhos relacionados, nela o símbolo ✓ indica que a solução avaliada apresenta determinada capacidade de maneira completa, enquanto o símbolo ✳ indica a parcialidade e ✗ a ausência da mesma.

	Trabalho Referenciado				
	[Mehraghdam]	[Dräxler]	[Ocampo]	[Gil-Herrera]	[Wang]
Modelo de Especificação de SFC	Gramática Livre de Contexto	Grafo de Serviço (Explícito)	Grafo de Serviço (Implícito)	Grafo de Serviço (Implícito)	Grafo de Serviço (Implícito)
Solução/Formulação do Problema	Heurística Redutiva	Análise de Pareto	ILP	Pesquisa Tabu	Algoritmo Hasse e Algoritmo de Prioridades
Função Objetivo	Estática	Estática	Estática	Estática	Estática
Métricas de Avaliação	Taxa de Tráfego	Taxa de Tráfego Requisitos de Recursos Tamanho da SFC	Requisitos de Banda	Requisitos de Banda	Prioridade
Suporte a Criação de Ramificações	✳	✓	✓	✓	✓
Suporte a Ordenamento Parcial	✓	✓	✓	✓	✓
Suporte a Dependência de Ordem	✗	✗	✓	✓	✓
Suporte a Dependência de Acoplamento	✗	✗	✳	✳	✳

Tabela 1. Sumarização de Características das Soluções de Composição

Os modelos de especificação de cadeias de serviço indicam o nível de dependência do operador de rede na etapa de composição. Modelos que especificam explicitamente uma topologia parcial do serviço estão nos trabalhos [Mehraghdam et al. 2014] e [Dräxler and Karl 2017]. Já modelos implícitos, onde apenas possíveis relacionamentos entre funções de rede são descritos ou inferidos automaticamente estão presentes em [Ocampo et al. 2017], [Gil-Herrera and Botero 2017] e [Wang et al. 2017]. Porém, a automatização completa da etapa de composição pode resultar em limitações indesejadas que incidem sobre os operadores de rede, ou seja, necessidades e conhecimentos específicos das funções que compõem o serviço e/ou do ambiente onde este será integrado (*e.g.*, fixação de posicionamento, pontos de monitoramento) são desconsiderados. Por outro lado, os modelos topológicos explícitos também apresentam limitações de especificação de algumas estruturas topológicas importantes, a exemplo disto estão a ausência de suporte para a definição de ramificações não terminais ([Mehraghdam et al. 2014]) e de dependências entre funções de rede ou domínios de execução ([Mehraghdam et al. 2014] e [Dräxler and Karl 2017]).

Como todos os trabalhos analisados utilizam funções objetivo estáticas, as soluções são otimizadas para obterem os resultados mais adequados (sejam estes locais ou globais) para um grupo fechado de métricas. Os trabalhos [Mehraghdam et al. 2014] e [Dräxler and Karl 2017] realizam a busca pela melhor composição dentro das combinações possíveis de seus segmentos parcialmente ordenados. Enquanto a primeira simplesmente ordena tais elementos a fim de minimizar a taxa de tráfego agregada, a segunda utiliza uma análise de Pareto para identificar as topologias que maximizam o custo/benefício entre as três métricas consideradas. Já os trabalhos [Ocampo et al. 2017], [Gil-Herrera and Botero 2017] e [Wang et al. 2017] não operam sobre a fixação e combinação de segmentos parcialmente ordenados, mas na criação das topologias por inteiro. Dessa forma, as soluções visam otimizar as métricas assistidas ao mesmo tempo que a topologia é formada, para isto diferentes abordagens podem ser adotadas, como a análise de cada transição através de PLI ([Ocampo et al. 2017]), a decisão da transição através de busca tabu ([Gil-Herrera and Botero 2017]) ou a análise de todas

as possíveis transições descritas em um diagrama Hasse ([Wang et al. 2017]).

As soluções de composição apresentadas no decorrer desta seção, independente de sua eficiência na determinação de resultados consistentes com seus objetivos, não permitem ao operador de rede requisitar avaliações personalizadas de serviços de rede no que diz respeito às suas funções objetivo. Apesar de haver indícios da extensibilidade de alguns métodos de composição, permitindo assim a avaliação de funções objetivo diferentes daquelas prefixadas nas soluções, não existe uma estratégia formal nem uma ferramenta genérica que possibilite que essa mudança seja feita sem a necessidade de manipulação de código fonte. Finalmente, como decorrência da construção de soluções estáticas de composição, os modelos de requisição de SFC apresentam, além de indicações da topologia parcial do serviço, apenas os valores de entrada para métricas de avaliação previamente definidas, não sendo prevista nenhuma forma de descrever as métricas em si e nem mesmo as operações de avaliação a serem aplicadas sobre elas.

3. *CUstomazable Service COmposing*

O *CUstomazable Service COmposing*[1] (CUSCO) é uma solução de composição de serviços de rede que preza pela flexibilidade de configuração e personalização de avaliação para gerar topologias absolutamente ordenadas válidas a partir de uma cadeia parcial e um conjunto de dependências previamente especificadas através de uma gramática livre de contexto. Essas informações são providas em conjunto com métricas e diretivas de avaliação em um documento estruturado pela linguagem de serialização de dados *YAML Ain't Markup Language* (YAML). Esta seção apresenta em detalhes o modelo de requisição adotado e a metodologia de composição executada pela solução.

3.1. Modelo de Requisição

O modelo de requisição YAML apresenta quatro objetos principais que contemplam metadados (`METADADOS`), além de informações sobre o serviço (`SERVICO`), funções de rede (`ESPECIFICACAO`) e função objetivo (`FUNCAO_OBJ`). Em especial, o objeto da função objetivo suporta a especificação de múltiplas métricas definidas como uma coleção de sub-objetos. Esses sub-objetos informam, além de uma identificação única, o peso da métrica para a avaliação global (um valor real [0;1], sendo o somatório deste atributo entre todas as métricas igual a um), um valor inicial de entrada, as operações de avaliação (operações matemáticas), a forma de atualização após cada iteração avaliativa e o objetivo de avaliação (maximização ou minimização). A função objetivo é aplicada à topologia inserida no objeto de serviço e é calculada considerando os dados de *benchmarking* das métricas e expectativas de divisão de tráfego (na ocorrência de cada ramificações na cadeia) presentes no objeto de especificação. A Figura 1 ilustra, em alto nível, as chaves que compõem o modelo de requisição adotado.

A topologia do serviço parcialmente ordenado deve ser construída através da gramática livre de contexto denominada *Service ChAin Grammar* (SCAG), desenvolvida como parte deste trabalho. A formalização da gramática proposta é realizada através da quádrupla SCAG = $(\nu, \tau, \rho, \varsigma)$. Nesse caso, ν representa o conjunto de símbolos não terminais (`INICIO`, `BLOCOOP`, `BLOCOOPR`, `BLOCOTP`, `ORDEMP`, `EXCECAO`, `RAMIFICACAOT`, `PRAMIFICACAOT`, `RAMIFICACAONT`, `RAMIFICACAONTINT`,

[1]Disponível em https://github.com/ViniGarcia/NFV-FLERAS

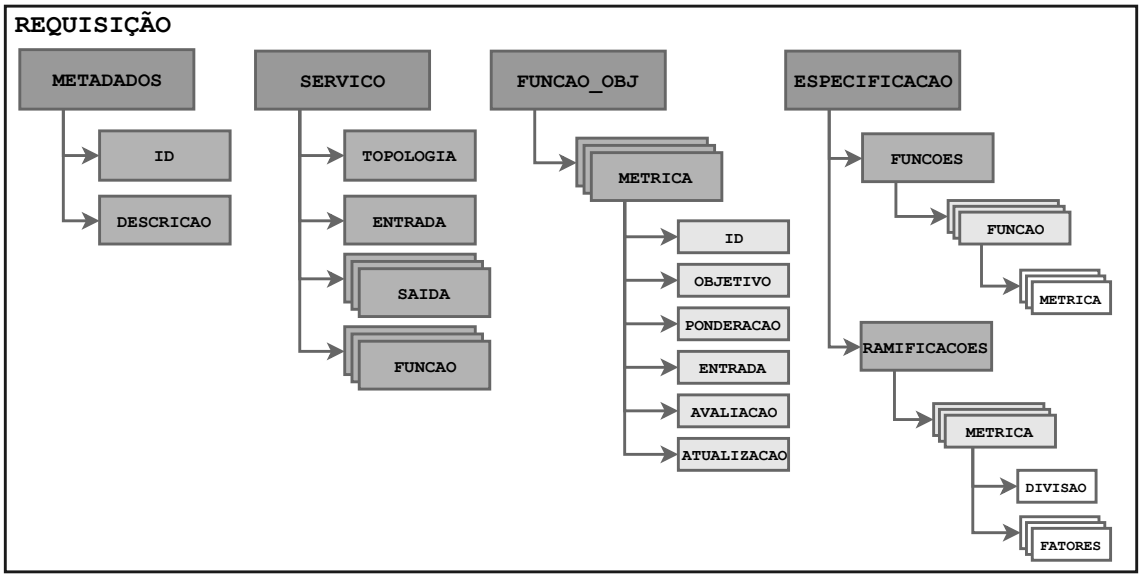


Figura 1. Modelo de Requisição CUSCO

PRAMIFICACAONT, PFUNCI, FUNCI, FUNC, NS, DOMINIO) e τ o conjunto de símbolos terminais (símbolos de controle - '<', '>', '{', '}', '(', ')', '[', ']', '/', '*', 'NE' - e símbolos variáveis - presentes nas regras de produção indicadas por FUNC, NS e DOMINIO) que formam a gramática. Esses símbolos são utilizados nas regras de produção definidas por ρ e apresentadas na Figura 2. Por fim, ς representa a regra de produção inicial da gramática (INICIO).

```
1 INICIO → 'NE' BLOCOOP
2 BLOCOOP → RAMIFICACAOT | RAMIFICACAONT | BLOCOTP BLOCOOP | BLOCOTP
    NS
3 BLOCOOPR → RAMIFICACAONTINT | BLOCOTP BLOCOOPR | BLOCOTP
4 BLOCOTP → ORDEMP | FUNCI
5 ORDEMP → '[' FUNCI PFUNCI ']' EXCECAO | '[' FUNCI PFUNCI ']'
6 EXCECAO → '(' FUNC FUNC ')' EXCECAO | '(' FUNC FUNC ')' | '(' FUNC
    FUNC '*' ')' EXCECAO | '(' FUNC FUNC '*' ')'
7 RAMIFICACAOT →  BLOCOTP '{' BLOCOOP PRAMIFICACAOT '}'
8 PRAMIFICACAOT → '/' BLOCOOP PRAMIFICACAOT | '/' BLOCOOP
9 RAMIFICACAONT → BLOCOTP '{' BLOCOOPR PRAMIFICACAONT '}' BLOCOOP
10 RAMIFICACAONTINT → BLOCOTP '{' BLOCOOPR PRAMIFICACAONT '}' BLOCOOPR
11 PRAMIFICACAONT → '/' BLOCOOPR PRAMIFICACAONT | '/' BLOCOOPR
12 PFUNCI → FUNCI PFUNCI | FUNCI
13 FUNCI → FUNC '<' DOMINIO '>' | FUNC
14 FUNC → 'FNC1' | 'FNC2' | 'FNC3' | ... | 'FNCn'
15 NS → 'NS1' | 'NS2' | 'NS3' | ... | 'NSn'
16 DOMINIO → 'DOM1' | 'DOM2' | 'DOM3' | ... | 'DOMn'
```

Figura 2. Regras de Produção da SCAG

A SCAG permite a criação de cadeias de serviço contendo tanto funções de rede absolutamente ordenadas, quanto em segmentos parcialmente ordenados delimitados pelos símbolos '[' e ']'. Sobre os segmentos parcialmente ordenados podem incidir dependências inseridas entre os símbolos '(' e ')', estas podem ser de ordem (precedência

em qualquer posição, sem o modificador '*') ou de acoplamento (precedência em primeira posição anterior, com o modificador '*'). Também, cada função de rede pode conter uma dependência de infraestrutura indicando, entre os símbolos '<' e '>', o domínio de execução da mesma. Além disso, ramificações terminais (*i.e.*, ramos irmãos) e não terminais (*i.e.*, com intersecção de ramos irmãos em uma função comum) podem ser especificadas entre os símbolos '{' e '}', com ramos separados pelo símbolo '/'. Finalmente, o modelo assume a existência de um único nó de entrada de dados e diversos nós de saída, além de encaminhamento unidirecional de tráfego. Todas as características relatadas são analisadas pela solução de composição em diferentes etapas do seu processamento.

3.2. Metodologia de Composição

A metodologia de composição CUSCO é dividida em dois procedimentos: expansão de topologias e avaliação de topologias. A expansão de topologias determina, através da resolução de ordenamentos parciais e reduções de ramificações, um conjunto de composições candidatas capazes de prover o serviço requisitado. Já a avaliação de topologias identifica e processa as métricas presentes na requisição, constrói funções parciais de avaliação e uma função objetivo geral e as utiliza para avaliar cada uma das composições candidatas, atribuindo a elas um Índice de Adequabilidade Topológica (IAT) que caracteriza uma composição candidata em relação às demais considerando a função objetivo geral. O IAT é então empregado no ranqueamento das composições candidatas, possibilitando que a solução sugira uma ou mais organizações topológicas a serem adotadas nas etapas seguintes da implantação do serviço.

3.2.1. Expansão de Topologias

O procedimento de expansão de topologias é subdividido em dois passos: resolução de ordenamentos parciais e redução de ramificações. Esses passos recebem como argumentos os dados fornecidos pelo objeto de serviço presente em uma requisição. No objeto os atributos `ENTRADA`, `SAIDA` e `FUNCAO` são empregados para distinguir os identificadores presentes na topologia requisitada (atributo `TOPOLOGIA`), além de serem diretamente usados na validação da mesma. O atributo `TOPOLOGIA`, por sua vez, fornece a cadeia de serviço parcialmente ordenada e informações extras que possibilitam a execução da expansão (*i.e.*, dependências, organização e ordenamentos topológicos), a execução desses passos resulta em um conjunto de composições candidatas absolutamente ordenadas que implementam o serviço requisitado.

A resolução de ordenamentos parciais é processada analogamente a uma permuta com restrições onde as restrições são as dependências de ordem e acoplamento. Dependências de ordem reduzem o número de possíveis posições de uma função de rede, sendo o tamanho desta redução relativa à quantidade de posições que sucedem aquela ocupada pela segunda função participante da dependência. Já uma dependência de acoplamento implica na fixação da primeira função na posição logo anterior a segunda, sendo ambas compreendidas como um bloco indivisível e tratadas como uma única função. A solução gera todas as permutas possíveis para um segmento parcialmente ordenado e verifica a validade das mesmas (*i.e.*, força bruta). Assim, a quantidade total de ordenamentos absolutos analisados para um segmento parcialmente ordenado é determinado pelo fatorial do número de funções de rede presentes no mesmo, enquanto a quantidade

de composições válidas ao final deste passo corresponde ao produto entre o número de ordenamentos absolutos válidos gerados em cada segmento processado.

A redução de ramificações consiste no processamento dos ramos de uma topologia absolutamente ordenada para detectar segmentos que podem ser alocados em uma seção comum antes do início ou após a conclusão de uma ramificação em análise. Ademais, segmentos equivalentes localizados em posições intermediárias dos ramos também podem ser unidos. Para isso, uma ramificação qualquer é decomposta em duas ramificações, uma de característica não terminal, estabelecida do ponto inicial da ramificação original até sua conclusão que ocorre antes do início do segmento a ser unificado (sendo a primeira função de rede deste segmento o ponto de intersecção dos ramos), e outra com a mesma característica da ramificação original, iniciada após o segmento unificado e perpetuada até a conclusão da ramificação original, ou seja, até os pontos de saída no caso de uma ramificação terminal ou até o elemento operacional de intersecção no caso de uma ramificação não terminal.

A solução realiza a redução de ramificações em segmentos iniciais e finais dos ramos, desconsiderando o caso de segmentos intermediários. As composições candidatas retornadas pela resolução de ordenamentos parciais são examinadas exaustivamente para identificar possíveis reduções em suas ramificações. Uma ramificação pode ser reduzida quando existem funções comuns a todos os seus ramos (tanto em tipo quanto em posição), além de, quando presentes, manifestarem dependências de infraestrutura compatíveis. Assim, a quantidade de composições candidatas geradas pela redução de uma ramificação é igual ao somatório do total de funções de rede presentes nos segmentos comuns máximos inicial e final (*i.e.*, maior conjunto de funções análogas entre os ramos), já que estes podem ser decompostos em subsegmentos menores. Também, a quantidade de novas composições resultantes desse passo quando aplicado a uma composição candidata é igual ao produtório do tamanho dos segmentos comuns máximos de todas as ramificações reduzidas, uma vez que estas reduções podem ocorrer simultaneamente.

3.2.2. Avaliação de Topologias

A avaliação de topologias é o segundo procedimento executado pela solução de composição e consiste do processamento das composições candidatas de um serviço de rede em vista das métricas, operações, ponderações e demais características informadas no objeto de função objetivo presente na requisição de um serviço. Esse procedimento é dividido em três passos: o primeiro compreende a definição de funções parciais e a avaliação das funções de rede e demais estruturas e nós especificados nas composições candidatas; o segundo realiza a depuração dos resultados parciais, os padronizando em um intervalo comum de valores; por fim, o terceiro passo utiliza os resultados padronizados para gerar um IAT destinado a cada composição candidata avaliada, possibilitando o ranqueamento das mesmas. As notações que formalizam as operações realizadas durante este procedimento são sumarizadas pela Tabela 2.

O primeiro passo trata da avaliação individual das composições candidatas, iterando sobre as posições de suas topologias para agregar resultados de avaliação. Para isso, inicialmente, funções parciais precisam ser construídas. Uma função parcial destina-se à avaliação das composições candidatas em relação a uma única métrica requisitada. Dessa

Notação	Descrição
CET	Conjunto de composições candidatas (topologias absolutamente ordenadas) gerado durante o passo de expansão de topologias.
CC	Composição candidata composta por funções de rede e nós de borda absolutamente ordenados.
FNC	Função de rede descrita em um documento de requisição.
P_{mtc}	Peso da métrica ”mtc”.
E_{mtc}	Valor de entrada da métrica ”mtc”.
FNC_{mtc}	Valor da métrica ”mtc”informado para a função de rede ”FNC”.
$A_{mtc}(E_{mtc}, FNC_{mtc})$	Operação de avaliação da métrica ”mtc”aplicada sobre a sua entrada ”E”e o valor da mesma métrica colhido para uma função ”FNC”.
$AVA_{mtc}^{CC\ CET}$	Resultado da avaliação da função parcial referente a métrica ”mtc”para a composição candidata ”CC”pertencente a ”CET”.
$DA_{mtc}^{CC\ CET}$	Distância absoluta máxima entre o resultado da avaliação da composição candidata ”CC”e o conjunto de resultados de avaliação das composições candidatas existentes em ”CET”pela métrica ”mtc”.
$NORM_{mtc}^{CC\ CET}$	Normalização do resultado da avaliação da composição candidata ”CC”em relação ao conjunto de resultados de avaliação das composições candidatas existentes em ”CET”pela métrica ”mtc”.
$RPP_{mtc}^{CC\ CET}$	Resultado parcial padronizado da composição candidata ”CC”em relação ao conjunto de resultados de avaliação das composições candidatas existentes em ”CET”pela métrica ”mtc”.
IAT_{CC}^{CET}	Índice de adequabilidade topológica da composição candidata ”CC”em relação ao conjunto de composições candidatas ”CET”.

Tabela 2. Notações de Formalização da Avaliação de Topologias

forma, cada uma das métricas presentes no objeto de função objetivo de uma requisição é transcrita para uma função parcial a ser calculada. O cálculo das funções parciais retornam valores agregados brutos que consistem do somatório dos resultados de avaliação de cada função operacional, estes devem estar disponíveis para todas as composições candidatas do conjunto retornado pelo primeiro procedimento antes da execução próximo passo. A Equação 1 generaliza o cálculo e o resultado de uma função parcial aplicada sobre funções de rede de uma composição candidata.

$$AVA_{mtc}^{CC\ CET} = \sum_{FNC}^{CC} A_{mtc}(E_{mtc}, FNC_{mtc}) \tag{1}$$

Ainda, o cálculo de uma função parcial é sucedido pela atualização das entradas e pela definição, caso necessária, de novos fluxos de execução, concluindo uma iteração avaliativa. Logo após a avaliação das métricas, estas são atualizadas de acordo com a sua própria operação de avaliação, substituindo a entrada pelo seu resultado, ou, da mesma forma, pelo resultado de avaliação de outra métrica presente na função objetivo. Também, uma terceira opção é a manutenção de um valor de entrada estático. Em seguida, o fluxo padrão de iterações, que consiste da avaliação de posições subsequentes da cadeia de serviço, pode ser alterado na eminência do início de uma ramificação. Nesse caso, a solução avalia todos os ramos (*i.e.*, do ponto inicial até o ponto de intersecção, onde ocorre uma agregação das entradas, ou nó de saída, encerrando a avaliação) através de uma busca em profundidade. Com isso, a entrada das métricas é atualizada por uma

operação de divisão de tráfego sobre fatores determinados para cada ramo (objeto de especificação), encaminhando estes valores ao início do novo fluxo de execução.

O segundo passo recebe os resultados provenientes da avaliação das composições candidatas pelas funções parciais de todas as métricas que compõem a função objetivo, os organizando no intervalo contínuo padrão [0;1]. A padronização do resultado de avaliação de uma métrica sobre uma composição candidata consiste em determinar a representatividade deste em relação aos demais resultados obtidos para a mesma métrica, mas provenientes de composições candidatas diferentes. Assim, o conjunto contínuo mínimo que compreende a totalidade dos resultados de avaliação de uma métrica, iniciado pelo resultado de Borda Inferior (BI) e encerrado no resultado de Borda Superior (BS) [BI;BS], deve ser mapeado ao intervalo contínuo [0;1] de forma que zero represente BI e um represente BS. Para isso, inicialmente, o intervalo [BI;BS] deve ser deslocado de modo que a sua origem se iguale a origem do intervalo objetivado (*i.e.*, zero), sendo necessária a transcrição dos resultados para o intervalo de distância absoluta máxima [0;$BS - BI$] utilizando a Equação 2.

$$DA_{mtc}^{CC\ CET} = AVA_{mtc}^{CC\ CET} - BI \tag{2}$$

Em seguida, o processo de normalização transcreve os valores do intervalo de distância absoluta máxima [0;$BS - BI$] de cada uma das métricas avaliadas para o intervalo padrão [0;1]. Isso é alcançado pelo cálculo da razão entre a distância absoluta verificada para as composições candidatas ($DA_{mtc}^{CC\ CET}$) e a distância absoluta máxima ($BS - BI$), como demonstra a Equação 3. Como exceção, a Equação 4 é empregada em cenários onde todas as composições candidatas apresentam o mesmo resultado para uma mesma métrica avaliada ($BI = BS$). O processo de normalização indica a proximidade de um dado resultado à borda superior, assim este pode ser naturalmente usado como o resultado parcial padronizado para métricas cujo objetivo é sua maximização (Equação 5). Por outro lado, métricas com o objetivo de minimização devem ter suas normalizações complementadas para a geração dos resultados parciais padronizados, como mostra a Equação 6, para que revelem a sua proximidade à borda inferior e possam ser analisados da mesma maneira e conjuntamente a métricas com objetivo de maximização.

$$NORM_{mtc}^{CC\ CET} = \frac{DA_{mtc}^{CC\ CET}}{DA_{mtc}^{CET}} \tag{3}$$

$$NORM_{mtc}^{CC\ CET} = 1 \tag{4}$$

$$RPP_{mtc}^{CC\ CET} = NORM_{mtc}^{CC\ CET} \tag{5}$$

$$RPP_{mtc}^{CC\ CET} = 1 - NORM_{mtc}^{CC\ CET} \tag{6}$$

Após a padronização dos resultados, os IAT das composições candidatas podem ser calculados. O IAT é um valor contido no intervalo contínuo [0;1] e indica a proximidade da avaliação geral da composição candidata ao resultado ótimo considerando todos aqueles recuperados no conjunto de composições avaliadas sobre todas as métricas requisitadas. Assim, uma composição adquire o maior IAT possível (*i.e.*, um) quando esta apresenta os resultados de borda inferior para todas as métricas com objetivo de minimização e resultados de borda superior para todas as métricas cujo objetivo é sua maximização.

De maneira prática, uma vez que os resultados já se encontram padronizados no intervalo de valores usado para o IAT, a sua definição consiste do somatório ponderado de todos os resultados parciais padronizados colhidos para uma determinada composição candidata (Equação 7), sendo a função objetivo adotada pela solução definida pela maximização (MAX) do IAT (Equação 8).

$$IAT_{CET}^{CC} = P_{mtc_1} * RPP_{mtc_1}^{CC\ \ CET} + ... + P_{mtc_n} * RPP_{mtc_n}^{CC\ \ CET} \tag{7}$$

$$MAX(IAT_{CC}^{CET}),\ \forall\, CC \in CET \tag{8}$$

Finalmente, uma vez que a solução de composição desenvolvida é hierarquizada em uma série de procedimentos e passos que, por mais que sejam interconectados, apresentam uma estrutura de execução independente e podem ser alterados ou substituídos desde que mantenham o modelo padrão de geração de dados. Em outras palavras, apesar de haver uma dependência relacionada à sequência de processamento entre procedimentos e passos, não há influência entre eles quanto ao algoritmo aplicado para a geração de seus dados de saída. Dessa forma, heurísticas podem ser utilizadas no primeiro procedimento sem que modificações sejam necessárias no segundo procedimento, da mesma forma que as funções parciais podem ser alteradas sem que haja necessidade de modificações na função objetivo geral da solução.

4. Experimentação

Para avaliar a solução proposta, um serviço de segurança, controle de conteúdo e balanceamento de carga foi desenvolvido como parte deste trabalho. O serviço usa um total de sete funções de rede virtuais, sendo duas implementadas em *Click Modular Router* (Filtro de Protocolo/Porta - `FPP`; Classificador de Tráfego - `CT`) e outras cinco implementadas em Python 3 (Inspetor de Assinaturas em Pacotes HTTPS - `IAPH`; Inspetor de Conteúdo de Pacotes HTTP - `ICPH`; Filtro de Marcação - `FM`; Sistema de Prevenção de Intrusão HTTP/S - `SPIH`; Balanceador de Carga - `BC`). A cadeia de serviço parcialmente ordenada especificada com o modelo SCAG (”`NE FPP CT { [IAPH FM SPIH] (IAPH FM*) / [ICPH FM SPIH] (ICPH FM*) } BC NS`”) balanceia requisições HTTP e HTTPS entre clientes e servidores, sendo as HTTP direcionados a análise de conteúdo e as HTTPS a análise de assinaturas, além de descartar tráfego com características diferentes de tais aplicações (TCP/IP, porta 80 para HTTP e porta 443 para HTTPS), com conteúdos ilegais ou reconhecidamente maliciosos pelo sistema de prevenção de intrusão.

A função objetivo empregue busca minimizar três métricas: taxa de tráfego, taxa de atendimento a requisições e tamanho da topologia. A taxa de tráfego consiste de um valor percentual, com entrada iniciada em 1 (100% do tráfego) e avaliação feita através da multiplicação desta pela taxa de tráfego de cada função iterada. A taxa de atendimento a requisições traduz-se em um número inteiro que indica a expectativa do máximo de requisições recebidas por segundo, sendo sua entrada igual a 100 e avaliação realizada através da diferença deste valor pela capacidade de atendimento a requisições por segundo das funções avaliadas. Por fim, o tamanho da cadeia assume um valor inteiro com entrada igual a 0, sendo uma unidade adicionada a ela a cada função iterada. A taxa de tráfego e capacidade de atendimento a requisições HTTP/S foram definidas por *benchmarking* das funções de rede. Além disso, todas as métricas apresentam o mesmo peso

(*i.e.*, ponderação 0,3333) e não houve preferência na divisão de tráfego no ponto de origem da única ramificação presente na topologia, especulando a equivalência entre tráfego HTTP e HTTPS que resulta em uma distribuição igualitária entre os seus ramos.

O processamento da cadeia de funções parcialmente ordenada do serviço de teste pela solução CUSCO resultou em nove composições candidatas. Dessas, três foram escolhidas para experimentação prática em um ambiente virtualizado real e são referenciadas pelos seus índices: IAT 0,666 ("`NE FPP SPIH CT {IAPH / ICPH} FM BC NS`"), IAT 0,500 ("`NE FPP CT {SPIH IAPH FM / SPIH ICPH FM} BC NS`") e IAT 0,327 ("`NE FPP CT {IAPH FM / ICPH FM} SPIH BC NS`"). A IAT 0,666 foi escolhida por apresentar o maior índice geral calculado, este oriundo de sua cadeia de tamanho mínimo (sete instâncias) e da menor taxa de tráfego constatada devido a antecipação da função `SPIH`. Entretanto, a IAT 0,666 obteve a pior taxa de atendimento a requisições visto que a existência de apenas uma instância da função `SPIH` no segmento principal pode se converter em um gargalo de processamento. A IAT 0,500 foi adotada por ser a composição com melhor índice geral cujo maximiza o RPP da taxa de atendimento a requisições, isso acontece pela replicação de instâncias das funções de rede nos ramos da topologia, ocasionando também, como consequência, o pior resultado de tamanho da topologia e resultados intermediários para a taxa de tráfego. Já a escolha pela IAT 0,327 é estimulada devido ao pior índice geral constatado, obtendo resultados intermediários para tamanho e taxa de atendimento a requisições, além do pior resultado de taxa de tráfego dada a postergação máxima das funções que descartam pacotes indesejados (`FM` e `SPIH`).

A implantação das composições candidatas selecionadas usou duas máquinas físicas para prover um ambiente virtualizado: a hospedeira dos clientes do serviço HTTP/S (8GB RAM DDR3, Core I3 4010U, Ubuntu 14.04 e virtualizador KVM); e a hospedeira do serviço de rede e dos servidores HTTP/S (8 GB RAM DDR3, Core I5 3330, Debian 8 e virtualizador KVM). Todas as máquinas virtuais (clientes, função de rede e servidores) foram instanciadas com 512 MB RAM e 1 núcleo virtual de processamento. As máquinas físicas são conectadas por interfaces Gigabit Ethernet e as virtuais por pontes, clientes e servidores encontram-se conectados na mesma rede /24. Nesse cenário são instanciadas dois servidores (ambos provendo serviços HTTP e HTTPS) e quatro clientes, dois para requisição de serviços HTTP e dois para HTTPS. Um cliente de cada aplicação é responsável por enviar requisições legítimas, enquanto o outro envia requisições maliciosas (pacotes anômalos de 1450 bytes, a uma taxa individual de 10Mbps) para indisponibilizar o serviço. Clientes e servidores são intermediados pelo serviço de teste. O ponto inicial da avaliação é o começo simultâneo do envio de requisições por parte dos clientes, já o ponto final remete a conclusão, com sucesso, de 5000 processos de requisições disparados pela ferramenta http-perf[2] em cada cliente legítimo.

A maior taxa de requisições HTTP/S atendidas por segundo, presentes na primeira e segunda colunas da Tabela 3, foi verificada para a composição candidata IAT 0,500. Isso acontece devido a presença de instâncias da função `SPIH` descartando tráfego malicioso no início dos ramos da topologia, além da alta capacidade de processamento de tráfego das funções anteriores a elas (`FPP` e `CT`, implementadas em *Click*), assegurando assim que funções com baixa capacidade de processamento não sejam oneradas com tráfego malicioso. Dessa forma, esse resultado confirma o melhor RPP da métrica de capacidade

[2]Disponível em https://www.npmjs.com/package/http-perf

	Maximização		Minimização	
	Taxa de Requisições HTTP Atendidas (req/s)	**Taxa de Requisições HTTPS Atendidas (req/s)**	**Taxa de Tráfego (Mbps)**	**Número de Instâncias de Funções de Rede**
IAT 0,666	2,4	2,5	37,9	7
IAT 0,500	20,4	13,0	59,6	9
IAT 0,327	0	0	**-**	8

Tabela 3. Resultados de Experimentação

de atendimento a requisições calculado pela solução de composição. A IAT 0,666 suporta o ataque recebido, entretanto a taxa de requisições atendidas é reduzida em relação a IAT 0,500. Essa situação decorre da existência de apenas uma instância da `SPIH` no segmento principal da cadeia, convertida em um gargalo durante a análise e descarte do tráfego malicioso. Já a composição IAT 0,327 aloca uma instância da `SPIH` na penúltima posição da cadeia, resultando no encaminhamento de tráfego malicioso para funções de baixa capacidade de processamento (`IAPH` e `ICPH`) e consequente indisponibilização do serviço.

A taxa de tráfego média no serviço (*i.e.*, quociente entre o total de tráfego processado pelas instâncias de função de rede e o tempo de execução até a conclusão da experimentação realizada), mensuradas para as composições candidatas que mantiveram o serviço disponível e exibidas na terceira coluna da Tabela 3, confirma a expectativa calculada pela solução de composição através do RPP da métrica taxa de tráfego. A composição IAT 0,666 adquire os melhores resultados para essa métrica devido à alocação das instâncias da função `SPIH` na segunda posição da cadeia, evitando que a classificação de tráfego (`CT`) ocorra sobre tráfego malicioso. Essa classificação, por outro lado, é necessária durante a utilização da composição IAT 0,500. Por fim, o número de instâncias de funções de rede virtualizadas presentes nas composições candidatas testadas, explícito na última coluna da Tabela 3, também confirma o RPP da métrica de tamanho calculada durante a determinação de seus respectivos índices. Dessa forma, é possível concluir que o comportamento operacional e características das cadeias de serviço no cenário e condições de teste demonstraram a viabilidade da solução CUSCO na composição e avaliação das mesmas.

5. Conclusão

A operacionalização de um serviço de rede no paradigma NFV passa pelo processo de implantação deste no ambiente virtualizado. Esse processo é constituído, principalmente, pelas etapas de composição, integração e agendamento de execução. Atualmente, a composição de cadeias de serviço é realizada por soluções estáticas e pouco personalizáveis em relação às suas funções objetivo e características de avaliação. Em vista disso, este trabalho apresentou o *CUstomazable Service COmposing*, uma solução genérica para a composição de cadeias de função de serviço parcialmente ordenadas orientada por métricas e operações de avaliação personalizadas. Por fim, um serviço de rede experimental foi composto e seus resultados testados em um ambiente virtualizado real, demonstrando a viabilidade da solução desenvolvida. Em trabalhos futuros objetiva-se generalizar e aplicar a metodologia CUSCO nas demais etapas de implantação.

Referências

Allybokus, Z., Perrot, N., Leguay, J., Maggi, L., and Gourdin, E. (2018). Virtual function placement for service chaining with partial orders and anti-affinity rules. *Networks*, 71(2):97–106.

Baek, H., Jang, I., Ko, H., and Pack, S. (2017). Order dependency-aware service function placement in service function chaining. In *Information and Communication Technology Convergence (ICTC), 2017 International Conference on*, pages 193–195. IEEE.

Dräxler, S. and Karl, H. (2017). Specification, composition, and placement of network services with flexible structures. *International Journal of Network Management*, 27(2):1963:1–20.

Faraci, G., Lombardo, A., and Schembra, G. (2016). A processor-sharing scheduling strategy for nfv nodes. *Journal of Electrical and Computer Engineering*, 2016:1:1–1:1.

Gil-Herrera, J. and Botero, J. F. (2017). A scalable metaheuristic for service function chain composition. In *Communications (LATINCOM), 2017 IEEE 9th Latin-American Conference on*, pages 1–6. IEEE.

Handley, M. (2006). Why the internet only just works. *BT Technology Journal*, 24(3):119–129.

Herrera, J. G. and Botero, J. F. (2016). Resource allocation in nfv: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3):518–532.

Kulkarni, S. G., Zhang, W., Hwang, J., Rajagopalan, S., Ramakrishnan, K., Wood, T., Arumaithurai, M., and Fu, X. (2017). Nfvnice: Dynamic backpressure and scheduling for nfv service chains. In *ACM Special Interest Group on Data Communication*, pages 71–84. ACM.

Mehraghdam, S., Keller, M., and Karl, H. (2014). Specifying and placing chains of virtual network functions. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, pages 7–13. IEEE.

NFVISG, E. (2012). Network functions virtualization: White paper. Technical report, European Telecommunications Standards Institute.

Ocampo, A. F., Gil-Herrera, J., Isolani, P. H., Neves, M. C., Botero, J. F., Latré, S., Zambenedetti, L., Barcellos, M. P., and Gaspary, L. P. (2017). Optimal service function chain composition in network functions virtualization. In *IFIP International Conference on Autonomous Infrastructure, Management and Security*, pages 62–76. Springer.

Quinn, P. and Nadeau, T. (2015). Problem Statement for Service Function Chaining - RFC 7498. Request for comments, Internet Engineering Task Force.

Rosa, R. V., Rothenberg, C. E., and Szabo, R. (2015). Vbaas: Vnf benchmark-as-a-service. In *Software Defined Networks (EWSDN), 2015 Fourth European Workshop on*, pages 79–84. IEEE.

Wang, Y., Li, Z., Xie, G., and Salamatian, K. (2017). Enabling automatic composition and verification of service function chain. In *Quality of Service (IWQoS), 2017 IEEE/ACM 25th International Symposium on*, pages 1–5. IEEE.

XXIV Workshop de Gerência e Operação de Redes e Serviços
SBRC 2019
Sessão Técnica 3

Um Mecanismo Adaptativo de Descoberta de Serviços para Internet das Coisas Centrado no Consumo Energético

Leonardo L. M. da Silva[1,2], Robson G. de Melo[1,2], Aldri Santos[1]

[1]Núcleo de Redes Sem-Fio e Redes Avançadas (NR2) – UFPR

[2]Núcleo de Redes Inteligentes e Sistemas Computacionais (RISC) – UNEMAT

{leoluan,robinho}@unemat.br, aldri@inf.ufpr.br

Abstract. *On the Internet of things, several techniques have been applied to perform the demand for services and guarantee the maintenance of the offered services. One of the popular techniques handles the usage of service repositories, which provide registration, update and service request messages. Though, the number of refresh messages exchanged between devices and repository impacts on the power of the devices, and consequently it decreases the availability of services. This paper introduces a mechanism for service discovery, called MADS, which adjusts the period of initialization of services taking into account the device's energy consumption. The operation routines show MADS reduces the number of service refresh messages required between device and repository. It also rises the service availability, and offers one rating of the availability indicator. In contrast to classic approaches, MADS gets to reduce up to 90% the exchange of refresh messages, and keeps up to 75% more available than others.*

Resumo. *A Internet das Coisas demanda pelo meios do gerenciamento da descoberta de serviços e pela garantia da manutenção dos serviços providos. Uma das possíveis abordagens empregadas na descoberta de serviços consiste no uso de repositórios de serviços, que suportam o registro, atualização e requisição de serviços. Contudo, o número de mensagens de atualização trocadas entre os dispositivos e o repositório causam impacto na carga energética dos dispositivos, e assim diminuem a disponibilidade dos serviços. Este trabalho propõe um mecanismo de descoberta de serviço, denominado MADS, que controla o período de atualização dos serviços de acordo com o consumo energético do dispositivo. As rotinas de execução mostram que o MADS diminui o número de mensagens de atualização de serviço trocadas entre dispositivo e repositório. Ele também tem melhorado a disponibilidade de serviços, além de oferecer uma classificação do indicador de disponibilidade. Em comparações com abordagens clássicas, MADS chega a reduzir até 90% a troca de mensagens de anúncio de serviços e ser até 75% mais disponível que outras soluções.*

1. Introdução

O paradigma da Internet das Coisas (IoT) se popularizou com o crescimento do número de dispositivos conectados à Internet [Atzori et al. 2010]. No ecossistema de IoT, uma grande rede pervasiva conecta uma quantidade sem precedentes de coisas e pessoas [Al-Fuqaha et al. 2015]. Porém, essa grande rede de conexões impõe novos desafios,

como, por exemplo, a descoberta de serviços. A descoberta de serviços é um dos fundamentos do ecossistema da IoT, pois por meio dela, usuários e objetos podem ter acesso aos serviços disponibilizados na rede por objetos até então desconhecidos.

O processo de descoberta de serviços demanda que dispositivos que ofereçam serviços na rede sejam encontrados e registrados de modo que seus serviços possam ser consumidos, e isso deve-se a algumas características dos dispositivos no ecossistema de IoT [Mohiuddin et al. 2014]. Primeiro, a maioria dos dispositivos são energeticamente limitados e, portanto, entram em modo de economia de energia, desligando seus rádios para economizar bateria. Segundo, em razão da mobilidade e dinamicidade da entrada e saída de dispositivos na rede ser grande, esses dois aspectos impõem aos mecanismos de descoberta de serviços a necessidade de atualização periódica da lista de serviços registrados, garantindo que esses serviços estejam sempre disponíveis para seus consumidores. A necessidade de atualização torna-se um problema devido ao grande número de mensagens trocadas no processo de anúncio, e causa grande consumo energético dos dispositivos e interfere na disponibilidade dos serviços registrados.

Os trabalhos na literatura têm empregados diferentes abordagens para esse problema. Em [Albalas et al. 2017] é utilizada uma abordagem baseada na *sequência de Fibonacci*. Nele os dispositivos anunciam seus serviços periodicamente em um *Diretório de Serviços*, que é responsável por registrar os serviços anunciados e atender requisições de busca por esses serviços. Para reduzir o número de mensagens de atualização, o intervalo de tempo entre anúncios é incrementado sempre que a carga energética do dispositivo decresce. Contudo, diante de uma possível recarga energética do dispositivo, o algoritmo estagna em um mesmo período de atualização, desconsiderando a nova condição energética. Outra limitação é a proporção fixa de aumento do período de atualização, não refletindo a situação energética de dispositivos com fonte energética abundante.

As abordagens para descoberta de serviço no ecossistema de IoT devem permitir que consumidores possam encontrar os fornecedores de serviços sem causar uma sobrecarga de mensagens na rede. Essa sobrecarga inviabiliza a utilização dos serviços por diversos fatores como o aumento do congestionamento de pacotes, diminuição do QoS da rede, entre outros [Aziez et al. 2017]. Outro aspecto importante trata da garantia miníma de disponibilidade do serviço, para que a medida que os consumidores desejem utilizar determinado serviços na rede, esses serviços estejam ativos e operantes com condições plenas de atendimento [Gigli and Koo 2011, Gubbi et al. 2013].

Este trabalho apresenta um mecanismo para solucionar o alto número de mensagens de anúncio na rede e combater o desperdício energético, que consiste no uso do *Mecanismo Adaptativo de Descoberta de Serviços (MADS)*, que ajusta periodicamente o tempo de atualização dos serviços de acordo com o consumo energético do dispositivo. O MADS também busca solucionar a falta de um indicador de disponibilidade dos serviços registrados por usar o período de atualização como índice de disponibilidade dos serviços registrados. Na medida em que um serviço apresenta um menor intervalo entre anúncios de atividade, maior será sua disponibilidade. A avaliação foi realizada com a implementação de rotinas de execução que mostram que o MADS diminui o número de mensagens de atualização de serviço trocadas entre dispositivo e repositório. O número de mensagens trocadas alterou de acordo com cenário de variação de carga energética. O MADS também foi eficaz na melhora dos indicadores de disponibilidade de serviços, garantido

que os serviços oferecidos na rede estejam disponíveis no momento de sua descoberta.

O restante deste trabalho está organizado da seguinte forma: A Seção 2 apresenta brevemente os conceitos e fundamentais sobre a descoberta de serviços em IoT. A Seção 3 apresenta os trabalhos relacionados. A Seção 4 detalha o MADS, bem como ilustra seu funcionamento. A Seção 5 apresenta a metodologia de avaliação do MADS e os resultados obtidos. Por fim, a Seção 6 apresenta as conclusões.

2. Visão Geral de Descoberta de Serviço em IoT

A descoberta de serviços tem por objetivo detectar de modo automático os serviços oferecidos em um escopo de rede [De et al. 2011]. No contexto de IoT, dispositivos chamados de **provedores de serviço** oferecem esses serviços. Os mecanismos de **descoberta de serviços** buscam, armazenam e organizam descrições dos serviços oferecidos na rede, indicando-os quando um solicitação é feita. Os **consumidores de serviços** utilizam essas descrições para terem acesso aos serviços de vários dispositivos desconhecidos. Existem inúmeros trabalhos na literatura que apresentam abordagens relacionadas à descoberta de serviços no contexto de IoT. O trabalho de [Broering et al. 2016] apresenta uma categorização dos mecanismos de descoberta de serviços baseada no escopo da busca *ao redor do nó consumidor*, no *escopo da rede*, e *em diretórios*, como retratado na Figura 1.

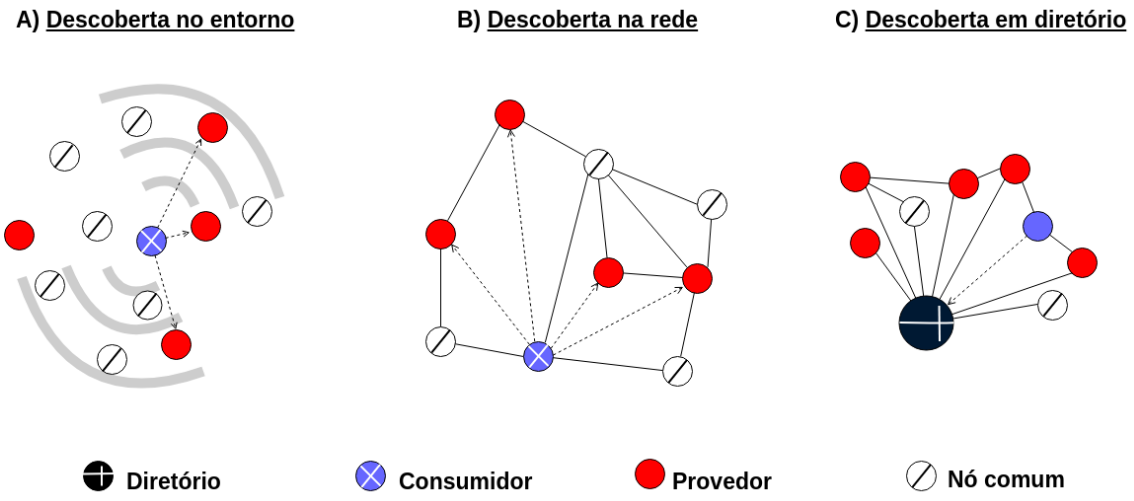


Figura 1. Escopos de descoberta de serviços

A categoria de descoberta de serviços no *entorno do nó consumidor* considera a proximidade espacial dos serviços e, portanto, está relacionado a mecanismos de rede sem-fio, em que o dispositivo consumidor observa em seu entorno como mostrado na Figura 1-A. Na categoria de descoberta de serviços no *escopo de rede*, as técnicas empregadas buscam serviços em toda a rede em que o consumidor está presente. As técnicas dessa categoria geralmente trabalham em escopo local, identificando todos os provedores de serviços da rede local através de mensagens *multicast*, como representado na Figura 1-B. Por fim, a categoria de *busca em diretório*, representada na Figura 1-C. Neste modelo estão classificadas as técnicas de descoberta e registro de serviços em diretórios, que podem ser dispositivos dedicados ou não. Assim, os diretórios de serviços são responsáveis pelos serviços propriamente dito e por receber as requisições de busca por eles.

3. Trabalhos Relacionados

Na literatura diferentes abordagens são apresentadas como forma de debater a descoberta de serviços no contexto de IoT. O trabalho de [Kaiser et al. 2016] aborda a descoberta de serviços usando *mDNS (Mobile DNS) e DNS-SD (DNS-Service Discovery)*. O objetivo é contrapor soluções baseadas em *multicast*, por não serem escaláveis e aprimorar o uso de DNS na descoberta de serviços. Eles propõem uma técnica de descoberta distribuída em *NS (Naming Servers)* sincronizados usando o algoritmo *Raft*. Os *NS* podem assumir a função de diretório de serviços e passam a ser chamados de *SNS (Scope Name Servers)*. Esses *SNS* respondem as requisições de registro e recuperação de serviços. Contudo, em redes maiores, com maior número de nós, o processo de sincronização dos *SNS* gera sobrecarga na rede.

Em [Ishaq et al. 2012], o objetivo consistiu na implantação e descoberta automática de serviços na rede. Eles propõem um mecanismo hierárquico autoconfigurável, em que o principal elemento é o *Gateway* de sensores. Esse *gateway* é responsável por buscar informações sobre todos os nós presentes na rede. Assim, ele envia uma mensagem *multicast* periodicamente a todos os nós da rede, requisitando informações para manutenção da lista de serviços descobertos. A informação é recuperada pelo protocolo *CoAP - Constrained Application Protocol* e os nós que oferecem serviços são registrados em um DNS local. O trabalho também propõe a integração entre *CoAP* e *HTTP* usando outro *gateway*, de mais alto nível, responsável por traduzir as mensagens entre os protocolos. O trabalho não apresenta análise detalhada dos resultados obtidos, porém, adianta que a abordagem *pull-based*, em que o nó detentor do serviço é passivo, tem problemas de escalabilidade, principalmente em relação a abordagem *push-based*, devido ao uso intenso de *multicast*.

O trabalho de [Helal and ElMougy 2015] apresenta uma solução distribuída e energeticamente eficiente utilizando *P2P-DHT* e *CoAP*. O objetivo é aprimorar a abordagem centralizada usando *gateway* dedicado para evitar a existência de um único ponto de falha. Uma arquitetura hierárquica onde alguns dos nós comuns da rede atuando como *Cluster Heads* mantêm registro de serviços disponibilizados por nós vizinhos. Nesta abordagem híbrida, nós comuns e *gateway* coexistem e dividem responsabilidades no processo de descoberta, resultando em uma arquitetura capaz de diminuir o consumo energético da rede. No entanto, os resultados apontam para uma dependência dos *gateway* dedicados para comunicação de um salto, sendo que ausência deles o consumo energético é maior e a taxa de resposta à requisições diminui.

Em [Kim et al. 2017], os autores propõem o uso de *DNS-SD* para a descoberta de serviços a fim de aprimorar o uso do DNS no processo de descoberta de serviços abrindo mão do *mDNS*, muito usado em conjunto com o *DNS-SD*. A solução apresentada no trabalho inclui a utilização da mensagem de resposta usando protocolo DHCP ao *gateway* da rede, com o nome do dispositivo que oferece o serviço e a lista de serviços oferecidos, que são registrados em um servidor DNS. Os dispositivos consumidores obtém a lista de dispositivos registrados do servidor DNS. Com isso, deve escolher um dispositivo específico e solicitar ao servidor DNS a lista de serviços oferecidos pelo dispositivo escolhido. Ao obter a lista de serviços do dispositivo, o consumidor faz uma requisição *CoAP* ao dispositivo servidor solicitando um de seus serviços. Essa proposta tem resultados conceitualmente interessantes devido ao modelo de interação entre dispositivos e na utilização do protocolo *DNS-SD*. Porém, apresenta limitações na manutenção de serviços,

dado que os dispositivos consumidores precisam escolher um dispositivo sem nenhuma descrição de seus serviços além do valor semântico do nome do dispositivo servidor.

O trabalho de [Barreto et al. 2017] apresenta uma abordagem considerando o contexto da aplicação. Para diminuir o número de dispositivos que oferecem um determinado serviço, considera o contexto dos serviços disponíveis. Esse contexto se refere à localização geográfica e o histórico de uso. Eles partem da hipótese que, ao diminuir o número de dispositivos resultantes do processo de descoberta, reduziria-se o número de mensagens trocadas entre o consumidor e os possíveis provedores. Embora a hipótese provou-se verdadeira, visto que os resultados apontam para uma diminuição no número de mensagens trocadas, o tempo de resposta aumentou devido ao refinamento da descoberta pelo mecanismo de contextualização baseado na localização e no histórico de uso do consumidor.

Um mecanismo de descoberta de serviços deve ser eficiente na troca de mensagens com os dispositivos provedores para evitar o gasto desmedido de energia com registros e atualizações. Isso deixa de lado o primeiro objetivo do serviço: ser útil aos clientes. Da mesma maneira, esse mecanismo deve garantir a disponibilidade dos serviços registrados, pois o oferecimento de serviços que não estão disponíveis também causam desperdício energético, além de tornar o mecanismo de descoberta ineficiente. Por isso, um número elevado de mensagens trocadas durante o processo de anúncio do serviço é um sério problema no ecossistema de IoT [Meshkova et al. 2008].

O trabalho de [Albalas et al. 2017] trata o número elevado de mensagens com um algoritmo para adaptação do período de atualização adequado a cada dispositivo usando uma abordagem baseada na *algoritmo de Fibonacci*. Assim, os provedores de serviço aumentam o período de atualização ao longo da operação do dispositivo à uma taxa de diminuição baseada na sequência de *Fibonacci*. Espera-se que o período de atualização aumente em decorrência da redução da carga energética, logo, reduzindo o número de mensagens trocadas para o anúncio de serviços oferecidos na rede [Albalas et al. 2019]. No entanto, essa solução só pode ser considerada adaptativa no sentido de realizar ajustes em tempo de operação no período de atualização, sem considerar variáveis importantes [Khalil et al. 2018]. No caso de uma possível recarga energética do dispositivo, o algoritmo ficaria estagnado em um mesmo período de atualização, desconsiderando a nova condição energética do dispositivo. Outra questão é a proporção fixa de aumento do período de atualização, o que não reflete a situação energética de dispositivos com fonte energética abundante.

4. MADS: Mecanismo Adaptativo de Descoberta de Serviços em IoT

Esta seção descreve o *Mecanismo Adaptativo de Descoberta de Serviço* (MADS) para o ecossistema de IoT. O MADS atua com dois objetivos principais: *i)* diminuir o número de mensagens trocadas na descoberta de serviços e *ii)* fornecer um índice de disponibilidade de cada serviço descoberto. A exigência de atualização periódica gera um aumento no número de troca de mensagens, um alto consumo energético e consequentemente reduz a disponibilidade do serviço, tendo em vista que aumentam as chances do dispositivo provedor de serviço ficar indisponível por falta de energia.

No ecossistema de IoT, o processo de descoberta de serviço acontece de maneira diferente do tradicional [Cirani et al. 2014]. Os dispositivos não têm acesso a fonte energética, ou seja, tem carga energética limitada e por conta disso, entram em modo de eco-

nomia de energia quando ficam ociosos ou quando a carga energética está acabando. Esse comportamento diferencia o processo de descoberta de serviços dos métodos aplicados por ferramentas de busca tradicionais, por exemplo, que têm acesso ilimitado a energia e aos provedores de serviço, já que esses não possuem restrição energética. Contudo, na IoT, a maioria das soluções possuem uma abordagem diferente, na qual os dispositivos que oferecem um serviço são responsáveis por anunciar periodicamente o serviço oferecido na rede ou em um diretório específico. Esses dispositivos são chamados de *Agentes de Serviços (AS)*, e o diretório específico é denominado *Diretório de Serviços (DS)*. No *MADS*, o *AS* sempre anuncia seus serviços em um diretório específico e o *DS* mantém uma lista com todos os serviços e o *AS* responsável por cada serviço registrado.

O *MADS* utiliza uma abordagem adaptativa a fim de diminuir o número de mensagens trocadas durante o processo de descoberta de serviços. Nessa abordagem, o período de atualização de um serviço é ajustado em função da variação de carga energética do *AS* responsável, em relação ao último anúncio realizado. Quando o período para o próximo anúncio é ajustado em função da variação de carga energética, o processo de descoberta considera as condições do *AS*, garantindo maior tempo de operação do serviço. Caso o agente receba muitas requisições de serviço, a variação de carga energética é maior e, o período para o próximo anúncio é maior, reduzindo o número de troca de mensagens.

Em relação ao fornecimento do índice de disponibilidade de serviços, o *DS* realiza um procedimento de classificação dos serviços registrados. Essa classificação é feita através da ordenação crescente, do período de atualização informada em cada anúncio realizado pelo *AS*. O *DS* mantém os registros de serviços ordenados para que o solicitante da lista de serviços seja capaz de decidir qual serviço requisitar, de acordo com o índice de disponibilidade do serviço. Desse modo, evita-se que o solicitante faça requisição de serviços que estejam indisponíveis ou desatualizado. No *MADS*, a probabilidade de um serviço estar disponível é maior quando o período de tempo até a próxima atualização é menor. Isso se deve ao fato de que o período de tempo é definido em relação a variação de carga energética do *AS* responsável pelo serviço. Da mesma maneira, quanto maior é o período de tempo para o próximo anúncio, maior foi o gasto energético do *AS* e maior é a possibilidade dele se tornar indisponível.

4.1. Arquitetura do MADS

A arquitetura do MADS compreende dois principais módulos: O **Módulo de Análise Energética (MAE)** e o **Módulo de Manutenção de Serviços (MMS)**, como ilustra a Figura 2. O Módulo de Análise Energética é executado no *AS*, enquanto o Módulo de Manutenção de Serviços opera no *DS*. Esses módulos se interagem por meio do *índice de disponibilidade* do serviço (**ID**) que é encaminhado do **MAE** para o **MMS** a partir da mensagens de anúncio. A estrutura arquitetural do MADS é sucinta e facilmente realizável no ecossistema de IoT. As regras para o funcionamento do mecanismo se baseiam nas especificações do protocolo *CoAP* e do padrão *CoRE Resource Directory* [Barreto et al. 2017]. Ambos protocolos são especificados pela IETF (*Internet Engineering Task Force*) e amplamente adotados como padrão da indústria da IoT.

O **MAE** é responsável por determinar o **ID** com base no cálculo da variação de carga energética do *AS* que é obtida a partir da entrada do valor da carga atual e da a carga anterior. Só então é realizado o ajuste do período do próximo anúncio de atualização do

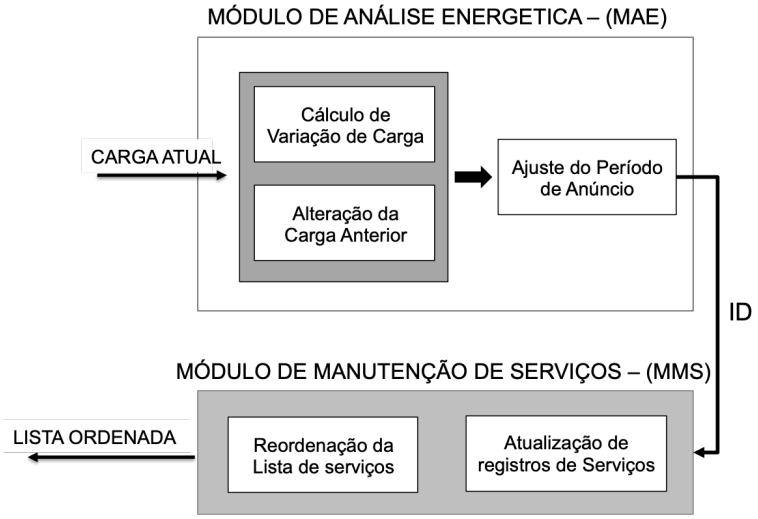


Figura 2. Arquitetura MADS

AS, usado para indicar o índice de disponibilidade. As variáveis para a operação desse módulo são:

- *Carga Energética Máxima* $(CMAX)$: corresponde ao capacidade máxima de carga energética do *AS*;
- *Carga Energética Imediata* (CI): corresponde ao valor da carga energética atual do *AS*;
- *Carga Energética Anterior* (CA): corresponde ao valor anterior da carga energética do *AS*;
- *Variação de Carga Energética* (ΔC): corresponde a diferença entre o *CA* e o *CI* de energia do *AS* como observado na Equação 1.

$$\Delta C = CA - CI \tag{1}$$

- *Prazo de Atualização* (PA): corresponde ao prazo da próxima atualização definida pela Equação 2.

$$PA = MomentoAtual + \Delta C \tag{2}$$

A operação do **MAE** pode ser dividida em duas etapas. A primeira consiste em calcular o valor da *CI* realizando uma medição da carga energética do dispositivo. Já segunda etapa realiza os procedimentos demonstrados no Algoritmo 1. Nele, a linha 2 realiza uma verificação do valor da variável *CA* e, caso esse valor ainda não estiver definido, trata-se do primeiro anúncio, que é o anúncio de registro no *DS*. Caso o valor da *CA* não esteja definido, este recebe o valor máximo de carga energética *(CMAX)* do *AS* (linha 3). Isso ocorre porque não existe anúncio anterior. Esse valor corresponde ao total de unidades de carga energética ou valor máximo de carga energética do *AS* (linha 3). O valor de *CMAX* é relativo e depende da capacidade de carga energética do *AS*.

A variável ΔC é inicializada com o valor 1 (um), ou seja, adota-se uma variação inicial de uma unidade de carga (linha 4). Após a inicialização, a variação de carga energética do *AS* é calculada (linha 6), com os valores de *CA* e *CI*. Nos casos em que

Algoritmo 1: Ajuste do Período de Anúncio

Entrada: $CA, CI, \Delta C, CMAX$
1 **início**
2 | **se** $CA = null$ **então**
3 | | $CA \leftarrow CMAX$
4 | | $\Delta C \leftarrow 1$
5 | **fim**
6 | **se** $CA > CI$ **então**
7 | | $\Delta C \leftarrow CA - CI$
8 | **fim**
9 | **se** $CA < CI$ **então**
10 | | $\Delta C \leftarrow CMAX - CI$
11 | **fim**
12 | $CA \leftarrow CI$
13 **fim**

o valor de *CA* é maior do que o valor de *CI*, ou seja, a carga energética do dispositivo diminuiu (linha 6), o valor da ΔC é igual a diferença entre os valores de *CA* e *CI* (linha 7). Nas situações em que o valor de *CI* é superior ao valor de *CA* (linha 9), deste modo, a carga energética do *AS* aumentou em relação à última medição, o valor de ΔC é igual à diferença do valor de *CMAX* e o valor de *CI* (linha 10). Assim, a variação será ajustada em situações de recarga e descarga energética. Na linha 12, o valor de *CA* é atualizado com o valor de *CI*, e passa a ser o referencial para o cálculo da variação no próximo anúncio. Essa atualização no valor de *CA* é importante para manter o valor de ΔC relativo à última medição de carga do *AS*. O valor de ΔC encontrado é importante, pois ele representa o período de tempo para o próximo anúncio, e isso define o *Prazo de Atualização (PA)*. Assim, o *AS* responsável pelo serviço têm definido o próximo momento de anúncio. Além de definir o *PA*, o ΔC define o *índice de disponibilidade* (**ID**) do serviço, que é enviado na mensagem de anúncio ao *DS*.

O **MMS** realiza a classificação de serviços com base no índice de disponibilidade recebido na mensagens de anúncio do **MAE**. A mensagem de anúncio do serviço é enviada por um canal seguro de comunicação. A principal função do **MMS** consiste em classificar os serviços registrados e responder às requisições dos dispositivos que solicitam a lista de serviços do repositório. Essa classificação promove os serviços com maior disponibilidade. Nesse módulo, dois procedimentos acontece: *i) atualização do registro de serviço anunciado*, e *ii) reordenação da lista de serviços* de acordo com o índice de disponibilidade do serviço. A primeira etapa de operação do módulo atualiza o registro do serviço no *DS*. Este procedimento ocorre quando o *AS* anuncia o serviço, e o *DS* recebe a mensagem de anúncio que contém o índice de disponibilidade do serviço. O *DS* então encontra o registro do serviço anunciado e atualiza o índice de disponibilidade. Essa etapa é fundamental à manutenção dos serviços registrados. Os registros no *DS* devem estar atualizados para reproduzir a situação do *AS* responsável por cada serviço.

Feita a atualização de registro do serviço anunciado, a lista de serviços mantida pelo *DS* deve ser reordenada. A reordenação da lista é feita em ordem crescente do índice de disponibilidade de cada serviço. Os serviços com o menor valor de índice de disponibilidade têm um período de anúncio menor, e portanto, tiveram menor gasto energético desde o anúncio anterior. Esse menor gasto energético pode ser referente a um menor número de requisições de serviços atendidos, ou referente à maior capacidade de carga energética do *AS* responsável pelo serviço. O resultado da reordenação dos serviços re-

gistrado é a lista ordenada de serviço. Essa lista colabora para a resolução do problema da falta de um indicador de disponibilidade dos serviços registrados. Em posse dessa lista, os consumidores de serviços podem requisitar os serviços no topo da lista, com maior possibilidade de obterem resposta mais confiáveis e disponíveis.

4.2. Fluxo de funcionamento do MADS e do AFT

O funcionamento dos módulos do MADS contribuem para a diminuição do número de mensagens de anúncio e para a geração do *indicador de disponibilidade* de cada serviço. A análise energética realizada no *AS* é importante para definir *períodos de anúncios* compatíveis com o gasto energético do dispositivo e para gerar o contexto de execução de onde é obtido o índice de disponibilidade, fundamental para gerar o indicador de disponibilidade de cada serviço do *DS*. A Figura 3 ilustra um esquema com o fluxo de funcionamento da *MADS* no *AS*, bem como um exemplo de operação do mecanismo *AFT (Adaptive Fibonacci-based Tuning Protocol)* de descoberta de serviços proposto por [Albalas et al. 2017]. O esquema é dividido em 3 camadas. Cada camada representa um fluxo de operação. A camada superior ilustra o fluxo de operação do *MADS*. A camada inferior representa o fluxo de operação do mecanismo *AFT*, por fim a camada central demonstra as operações comuns à ambas as abordagens.

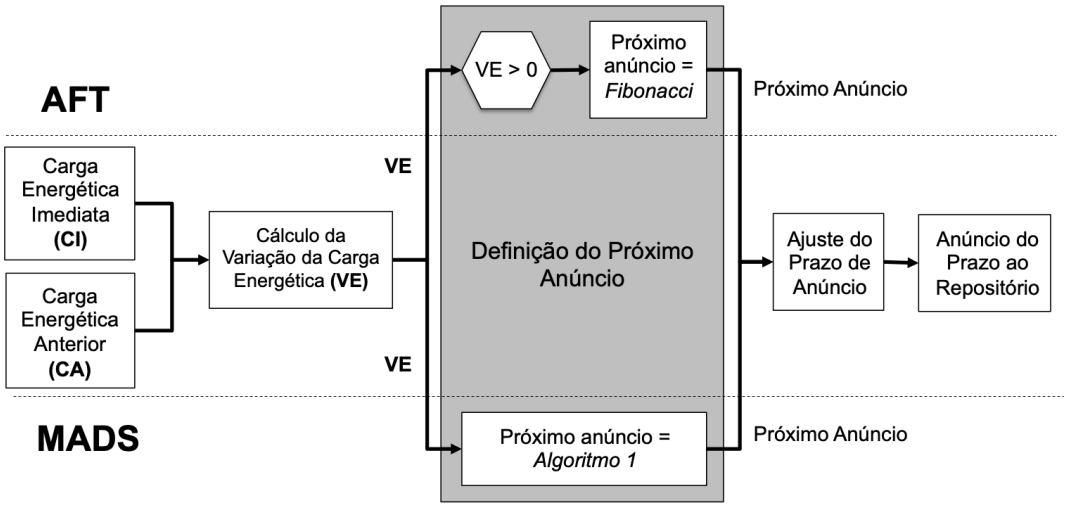


Figura 3. Fluxo de funcionamento - MADS x AFT

As duas primeiras operações determinam os valores de *CA* e *CI*. A terceira operação realiza o cálculo da valor de ΔC, correspondente a variação de carga. Essas 3 operações são comuns nos dois mecanismos. A seguir, na fase de definição do próximo anúncio, o fluxo de operações se diferenciam em cada mecanismo. Para definir o próximo anúncio, o mecanismo *AFT* verifica se houve variação na carga energética e, caso positivo, calcula um novo período de anúncio de serviço usando a *Sequência de Fibonacci*. Nessa fase, o valor da variação de carga energética é utilizado somente para verificar se houve diminuição na carga energética do dispositivo. Durante a mesma fase, no *MADS*, o período para o próximo anúncio do serviço é definido diretamente pelo valor da variação da carga. A principal diferença de operação dos dois mecanismos é a maneira como o valor da variação de carga energética é utilizado. No *MADS*, o período de anúncio de serviço é ajustado em razão do valor de variação de carga energética.

Definido o período para o próximo anúncio de serviço, o fluxo converge novamente na operação de ajuste do prazo de anúncio. Essa operação consiste no ajuste do novo prazo que o dispositivo tem para enviar uma nova mensagem de anúncio ao repositório de serviços. A última operação, também comum aos dois mecanismos, é o anúncio do prazo de atualização ao repositório de serviços. Nessa operação, o dispositivo envia uma mensagem de atualização ao repositório, contendo o novo prazo para atualização.

5. Avaliação

Esta seção descreve a metodologia de implementação e avaliação de desempenho do do MADs para validar a sua eficácia. Também é feita uma análise comparativa com o mecanismo AFT (*AFT: Adaptive Fibonacci-based Tuning Protocol*), proposto por [Albalas et al. 2017]. Para realizar a comparação, rotinas foram implementadas utilizando a linguagem de programação *Python* na versão 3.44. A implementação foi feita representando o cenário de troca de mensagens de anúncio durante períodos de observação de 5, 10, 30 e 60 minutos, conforme parâmetros descritos na Tabela 1. Esses períodos de observação foram escolhidos de forma arbitrária, e representam o intervalo entre o registro e o desligamento do serviço.

Tabela 1. Configuração da simulação

Parâmetros	Valores
Intervalos de observação	5, 10, 30 e 60 minutos
Taxa de variação de carga energética	1-5, 6-10, 11-15 e 16-20 unidades de carga
Mensagens por anúncio	1

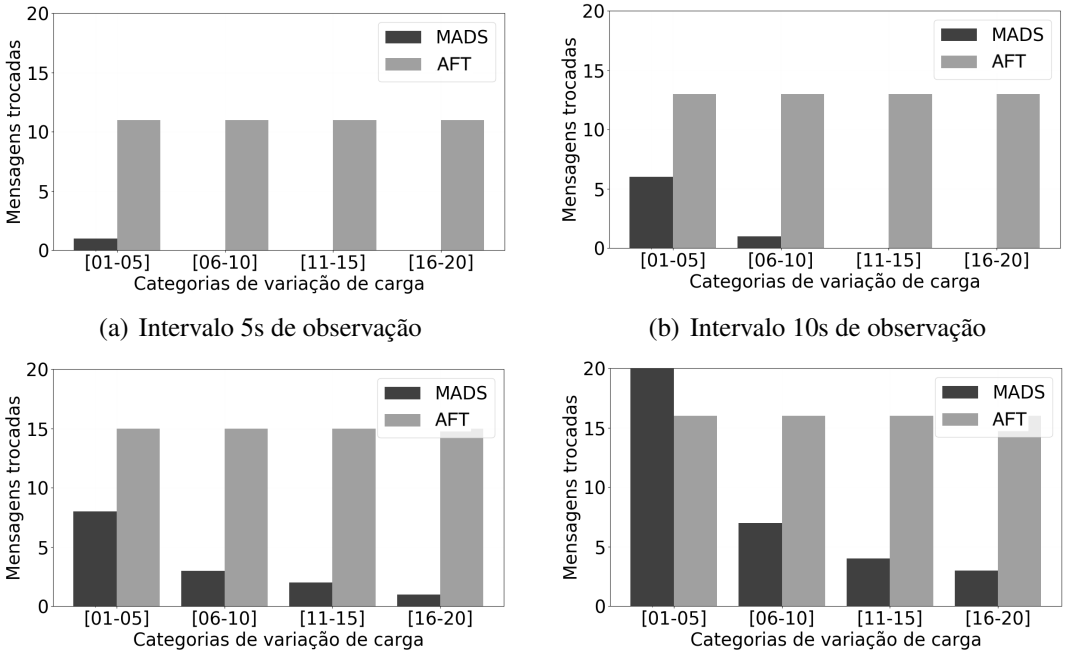
Durante a simulação, foram aplicadas taxas de variação de carga energética de 1-5, 6-10, 11-15 e 16-20 unidades por minuto. As unidades de carga energética são uma abstração do algoritmo para unidades de medida de carga energéticas reais, geralmente representadas em miliampére-hora (mAh). Foram executadas 30 rodadas de simulação, variando a carga de acordo com os valores anteriormente descritos.

Na análise do desempenho de ambos os sistemas, foram consideradas duas métricas: **número de mensagens trocadas (NMT)** e o **índice de disponibilidade dos serviços (IDS)**. O **NMT** representa as mensagens trocadas durante o processo de descoberta de serviços, o que consiste em toda a transação realizada com o solicitante de um serviço na rede e o provedor de serviço registrado no diretório de serviço associado. Já o **IDS** corresponde ao indicador de disponibilidade considerando que quanto menor o intervalo entre os anúncios de um provedor de serviço maior será sua disponibilidade devido sua demostração de atividade na rede.

5.1. Resultado: Número de mensagens de anúncio trocadas

O número de mensagens trocadas (NMT) para anunciar um serviço é importante no ecossistema de IoT. Uma maior quantidade de mensagens trocadas provoca maior consumo de energia dos dispositivos, devido ao uso frequente do rádio para realizar o envio dos pacotes. Os resultados obtidos nas experimentações são mostrados nos gráficos da Figura 4.

O gráfico 4(a) apresenta o desempenho do MADS comparado ao AFT no número de mensagens trocadas. O desempenho do MADS é superior na comparação durante



(a) Intervalo 5s de observação

(b) Intervalo 10s de observação

(c) Intervalo de 30s de observação

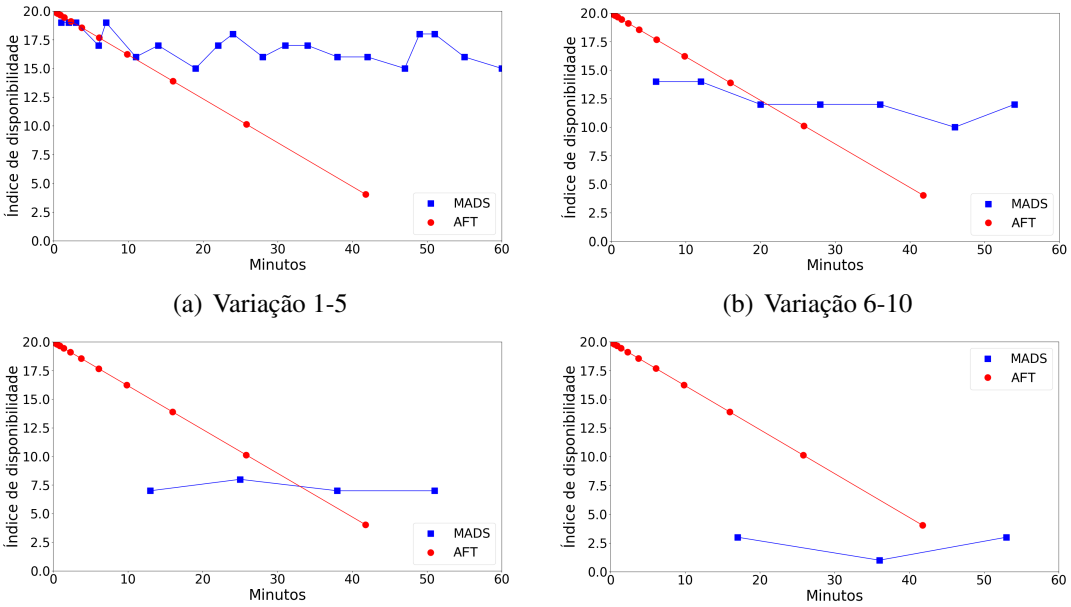
(d) Intervalo de 60s de observação

Figura 4. Avaliação de Mensagens de Anúncio Trocadas

5 segundos de anúncio do dispositivo, trocando uma mensagens somente na categoria de variação de carga energética entre 1 e 5. Nesse mesmo período, o AFT troca 11 mensagens em todas as taxa de variação. No gráfico 4(b), o desempenho do MADS continua superior em todas categorias de variação de carga energética, trocando mensagens somente nas categorias de variação entre 1 e 5, e 6 e 10. O número máximo de troca de mensagens durante 10 segundos atinge 6, enquanto o mecanismo AFT troca 13 mensagens em todas as categorias de carga energética. No gráfico 4(c), o MADS troca 8 mensagens com uma variação de carga energética entre 1 e 5, e apresenta decréscimo no número de mensagens trocadas nas outras categorias. Já o AFT troca 16 mensagens em todas as categorias de variação de carga energética. Já no gráfico 4(d), o desempenho do MADS na categoria de variação de carga energética entre 1 e 5 é inferior, trocando 20 mensagens, 2 a mais que o AFT. Porém, nas outras categorias de variação, o número de mensagens trocadas pelo MADS decresce, enquanto do AFT se mantém fixo em 18 mensagens. O desempenho do MADS em comparação ao mecanismo AFT, no geral, é superior. Nos 16 cenários de comparação, o número de mensagens trocadas pelo MADS é sempre inferior.

5.2. Resultado: Índice de disponibilidade do serviço

O índice de disponibilidade diz respeito à uma métrica importante na seleção de um serviço no ecossistema IoT. Nesse ecossistema, a mobilidade de um dispositivo e a falta de uma fonte energética ilimitada influenciam diretamente na disponibilidade de um serviço, dado que o dispositivo provedor pode se desconectar a qualquer momento. A disponibilidade de um serviço é relacionada, nesse trabalho, com a periodicidade com que um serviço realiza anúncios de atualização de serviço, ou seja, de sua atividade na rede. Na medida em que um serviço apresenta um menor intervalo entre anúncios, maior sua disponibilidade, dado que está mais ativo na rede.



(a) Variação 1-5

(b) Variação 6-10

(c) Variação 11-15

(d) Variação 16-20

Figura 5. Avaliação do Índice de Disponibilidade

Na Figura 5 são apresentados quatro gráficos comparativos entre o comportamento do índice de disponibilidade do MADS e do mecanismo AFT durante 60 minutos de observação. O gráfico 5(a) mostra o comportamento dos índices de disponibilidade baseados em variações entre 1 e 5 unidades de carga energética por minuto. O comportamento do MADS, apesar de irregular no sentido da variação do índice, apresenta um índice superior enquanto o valor do índice do mecanismo AFT diminui com o passar do tempo. No gráfico 5(b), o índice de disponibilidade do MADS cai, porém apresenta comportamento mais regular, e continua superior ao AFT na maior parte do tempo observado. No gráfico 5(c), a regularidade do índice do MADS melhora, porém o índice de disponibilidade cai novamente, e passa a ser inferior durante a maior parte do tempo, contudo apresenta um comportamento mais estável que o AFT, que se mantém em queda. Já no gráfico 5(d), o índice de disponibilidade do MADS é inferior durante todo o período de observação, sendo o valor mínimo alcançado bem próximo a zero.

No geral, na métrica da disponibilidade do serviço, o MADS é superior. Isso deve-se à abordagem adotada para definir os períodos de anúncio. O mecanismo AFT apresenta diminuição constante do índice de disponibilidade, e como pode ser observado em todos os gráficos da Figura 5, o último aviso dentro do período de observação acontece entre 40 e 50 minutos, longe do limite de 60 minutos, o que demonstra a chance crescente de que o serviço esteja indisponível no momento de uma requisição de serviço.

6. Conclusão

A descoberta de serviços é fundamental no ecossistema de IoT pois disponibiliza as funcionalidades, bem como os próprios serviços demandados pelos dispositivos conectados à rede. Para realizar a descoberta de serviço, várias proposta foram apresentadas na literatura. Na maioria dessas propostas, os dispositivos conectados a rede realizam anúncios periódicos de atualização dos serviços oferecidos ao diretório de serviços. A necessidade

de realização de atualização periódica gera um aumento no número de troca de mensagens, um alto consumo energético e consequentemente reduz a disponibilidade do serviço. Este trabalho apresentou o Mecanismo Adaptativo de Descoberta de Serviços (MADS) para diminuir o número de mensagens de atualização durante o processo de descoberta de serviço e para fornecer um indicador de disponibilidade de serviço.

As análises realizadas durante a execução das rotinas do mecanismos proposto avaliaram as métricas: *i) número de mensagens trocadas* e; *ii) valor do índice de disponibilidade*. Os resultados mostram que o MADS diminui o número de mensagens trocadas no processo de atualização e fornece um indicador de disponibilidade dos serviços. Além disso, o mecanismo contribui para a redução do consumo de carga energética dos dispositivos e auxilia na seleção do serviço mais disponível, colaborando para a melhora do processo de descoberta de serviço na IoT. Trabalhos futuros incluem o desenvolvimento de mecanismo para garantia de segurança durante a troca de mensagens de anúncio de serviços entre o provedor e o diretório de serviços, e o desenvolvimento de um procedimento de remoção de registros de serviços indisponíveis no diretório de serviços.

Referências

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

Albalas, F., Mardini, W., and Al-Soud, M. (2017). Aft - adaptive fibonacci-based tuning protocol for service and resource discovery in the internet of things. *IEEE Second International Conference on Fog and Mobile Edge Computing (FMEC)*.

Albalas, F., Mardini, W., and Yaseen, M. A.-S. Q. (2019). A topology–based performance evaluation for an adaptive tuning protocol for service and resource discovery in the internet of things. *Computing and Communication Workshop and Conference (CCWC) 2019 IEEE 9th Annual*.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *IEEE Computer Networks*, 54:2787–2805.

Aziez, M., Benharzallah, S., and Bennoui, H. (2017). Service discovery for the internet of things: Comparison study of the approaches. *IEEE 4th International Conference on Control, Decision and Information Technologies (CoDIT)*.

Barreto, F. M., de S. Duarte, P. A., Maia, M. E. F., de C. Andrade, R. M., and Viana, W. (2017). Coap-ctx: A context-aware coap extension for smart objects discovery in internet of things. *IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*.

Broering, A., Datta, S. K., and Bonnet, C. (2016). A categorization of discovery technologies for the internet of things. *ACM IoT'16 Proceedings of the 6th International Conference on the Internet of Things*, 6:131–139.

Cirani, S., Davoli, L., Ferrari, G., Léone, R., Medagliani, P., Picone, M., and Veltri, L. (2014). A scalable and self-configuring architecture for service discovery in the internet of things. *IEEE Internet of Things Journal*, 1:508 – 521.

De, S., Barnaghi, P., Bauer, M., and Meissner, S. (2011). Service modelling for the internet of things. *IEEE Federated Conference on Computer Science and Information Systems*, page 949–955.

Gigli, M. and Koo, S. G. M. (2011). Internet of things: Services and applications categorization. *Advances in Internet of Things*, 1:27–31.

Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Elsevier Future Generation Computer Systems*, 29:1645–1660.

Helal, R. and ElMougy, A. (2015). An energy-efficient service discovery protocol for the iot based on a multi-tier wsn architecture. *IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops)*, 40:862–869.

Ishaq, I., Hoebeke, J., Rossey, J., Poorter, E. D., Moerman, I., and Demeester, P. (2012). Facilitating sensor deployment, discovery and resource access using embedded web services. *IEEE Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*.

Kaiser, D., Waldvogel, M., Strittmatter, H., and Haase, O. (2016). User-friendly, versatile, and efficient multi-link dns service discovery. *IEEE International Conference on Distributed Computing Systems Workshops*, 36:146 – 155.

Khalil, K., Elgazzar, K., and Bayoumi, M. (2018). A comparative analysis on resource discovery protocols for the internet of things. *Global Communications Conference (GLOBECOM) 2018 IEEE*.

Kim, S., Lee, K., and Jeong, J. P. (2017). Dns naming services for service discovery and remote control for internet-of-things devices. *IEEE International Conference on Information and Communication Technology Convergence (ICTC)*.

Meshkova, E., Riihijärvi, J., Petrova, M., and Mähönen, P. (2008). A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *IEEE Computer Networks*.

Mohiuddin, J., Bhadram, V., Palli, S., and Koshy, S. S. (2014). 6lowpan based service discovery and restful web accessibility for internet of things. *IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI)*.

Monitoramento e Disseminação Cooperativa de Eventos Emergenciais Apoiado por Agrupamentos de Veículos

Everaldo Andrade[1], Kevin Veloso[1], Nathália Vasconcelos[1], Aldri Santos[2], Fernando Matos[1]

[1]Centro de Informática – Universidade Federal da Paraíba (UFPB)

[2]Departamento de Informática – Universidade Federal do Paraná (UFPR)

aldri@inf.ufpr.br, nathalia.vasconcelos@eng.ci.ufpb.br,

kevinveloso@gmail.com, everaldo.andrade@ppgi.ci.ufpb.br, fernando@ci.ufpb.br

Abstract. *Critical urban events occur at a random way and they need to be dealt with by public authorities quickly to maintain the proper operation of cities. The main challenges for an efficient handling of an event fall precisely in its random nature, and in the speed and accuracy of the notification of its manifestation to the authority. Moreover, the fact that this notification depends on human intervention aggravates this situation. The omnipresence of vehicles in urban environments, and their communication and monitoring capabilities, allow to employ VANETs to support the dissemination and handling of such events. Thus, this paper proposes MINUET, a system to support the monitoring and dissemination of urban events, which operates in an ad hoc vehicular network. It uses a cooperative-based strategy where vehicles are able to dynamically coordinate the monitoring and dissemination of multiple events in real time by the means of clusters. Results obtained by NS3 show the performance of MINUET regarding the real-time monitoring and dissemination of events detected in urban areas.*

Resumo. *Eventos urbanos emergenciais ocorrem de forma aleatória e precisam ser tratados por entidades competentes para manter o bom funcionamento das cidades. Os principais desafios para um tratamento eficiente recaem justamente na aleatoriedade no tempo ou no espaço, e na rapidez e acurácia da comunicação da ocorrência do evento a uma entidade responsável, sendo este último desafio agravado pelo fato desta comunicação depender de intervenção humana. A onipresença de veículos em ambientes urbanos, sua capacidade de comunicação e monitoramento, possibilitam o uso de VANETs como meio de auxílio para disseminação e tratamento destes eventos. Assim, este artigo propõe MINUET, um sistema de monitoramento e disseminação de eventos urbanos, que atua em uma rede veicular não estruturada. Ele é baseado em uma estratégia cooperativa, onde veículos coordenam dinamicamente o monitoramento e a disseminação de múltiplos eventos em tempo real através de agrupamentos. Resultados no NS3 mostram o desempenho do MINUET na detecção, monitoramento e disseminação em tempo real de eventos em zonas urbanas.*

1. Introdução

Nos últimos anos, os centros urbanos têm vivenciado um crescimento populacional sem precedentes, o qual impactará em vários aspectos do dia-a-dia da cidade, como por

exemplo na mobilidade urbana. Este aumento populacional dificultará cada vez mais a gestão da cidade por parte das entidades públicas ao agravar aspectos como a mobilidade urbana [Monreal 2018]. Dentre os vários problemas enfrentados pelos gestores das cidades, é possível destacar o tratamento de eventos emergenciais urbanos através da detecção, análise e monitoramento dos eventos [Memos et al. 2018]. Tais eventos podem ser incêndios, acidentes, assaltos, fuga de veículos, pessoas ou veículos procurados, bloqueios, manifestações, dentre vários outros. O tratamento rápido e eficiente destes eventos, através da colaboração entre agentes distribuídos no ambiente, torna-se primordial para garantir a segurança e a assistência dos indivíduos direta ou indiretamente envolvidos [Monajemi et al. 2018]. Além disso, uma resposta rápida garante um bom andamento do dia-a-dia da cidade. Por exemplo, ao detectar um bloqueio em uma avenida, a entidade pública de mobilidade deve agir rapidamente para desobstruir a via de modo a não criar longos e demorados congestionamentos em outras partes da cidade.

A infraestrutura de Tecnologias de Informação e Comunicação (TICs), existente em ambientes de Cidades Inteligentes, pode desempenhar um papel importante no tratamento destes eventos urbanos, onde a intercomunicação entre a infraestrutura urbana e as autoridades pode auxiliar no tratamento de tais eventos. Além disso, com a evolução de áreas como *Crowdsourcing* [Ma et al. 2018] e IoT [Hassan and Awad 2018], a integração e a cooperação de diferentes sistemas computacionais podem proporcionar o surgimento de novas estratégias no gerenciamento à respostas de emergências e monitoramento diário em zonas urbanas. Desta forma, é possível idealizar a criação de infraestruturas resilientes que garantam a continuidade das funcionalidades das cidades até mesmo em face de eventos catastróficos [Timashev 2017, Furutai and Kanno 2018]. Estas estratégias, normalmente fazem uso de fontes de captadores de informações (e.g. câmeras, sensores, *smartphones* e veículos) com o objetivo de auxiliar na tomada de decisões de outros sistemas responsáveis pelo tratamento de eventos específicos.

Os Sistemas de Transporte Inteligente (ITS) contribuem para a automatização de operações de deslocamentos dentro das Cidades Inteligentes. Em ITSs, veículos fazem parte de uma Rede Veicular não Estruturada (*Vehicular Ad Hoc Network* - VANET), através da qual se intercomunicam com outros veículos (*Vehicle-to-Vehicle* - V2V) e com a infraestrutura de redes da cidade (*Vehicle-to-Infrastrucutre* - V2I) para trocarem informações para os mais variados objetivos relacionados a segurança e entretenimento. Além disso, devido à sua onipresença, os veículos aparecem como uma alternativa óbvia para a coleta de dados multimídia nos centros urbanos [Wang 2018, Quadros et al. 2015b, Quadros et al. 2015a, Quadros et al. 2016] referente a eventos e para suportar *cooperação* e *coordenação* no monitoramento [Castro et al. 2017], uma vez que os mesmos não sofrem da limitação de recursos de processamento, armazenamento e energia que outras redes não estruturadas possuem [Khakpour et al. 2017]. Contudo, as VANETs impõem seus próprios desafios, tais como alta mobilidade [Silva et al. 2016], variação da densidade de tráfego, alta escalabilidade e variação topológica da rede [Khakpour et al. 2013, Macedo et al. 2014].

Embora estudos recentes têm mostrado estratégias ao gerenciamento de eventos em cidades, eles geralmente limitam-se ao tratamento de um evento específico, como por exemplo, inundações [Mousa et al. 2016], acidentes veiculares [Chou et al. 2017] ou condições de tráfego [Abhishek et al. 2016]. Outros trabalhos focam na disseminação de

conteúdos de vigilância [Huang et al. 2017] ou de colisões [Shi et al. 2017], porém em uma abordagem centralizada para a comunicação do evento. Contudo, a própria infraestrutura em cidades inteligentes poderia permitir uma colaboração através de *crowdsourcing* de veículos na detecção por imagens e disseminação de eventos emergenciais até uma entidade de controle pública externa com capacidade de processamento para reconhecer eventos e tomar as devidas providências de modo a otimizar as ações públicas.

Desta forma, este trabalho apresenta o modelo de um sistema baseado em agrupamentos cooperativos de veículos chamado MINUET (**M**onitorINg and Dissemination of **U**rban **E**ven**T**s), que opera sobre uma rede veicular híbrida para suportar a coordenação e o controle do monitoramento distribuído e dinâmico e disseminação sobre múltiplos eventos urbanos (e.g. acidentes, desastres e flagrantes) de modo a auxiliar as autoridades no reconhecimento destes eventos. No modelo proposto, ao se detectar um evento, os veículos se **auto-coordenam** para monitorar este evento o maior tempo possível de forma à entidade pública tomar a decisão mais eficaz para a sua resolução. Além disso, os veículos também coordenam uma comunicação cooperativa, onde múltiplos veículos transmitem dados de uma única fonte evento, e há a combinação dos dados transmitidos pelos agrupamentos de veículos na entidade destino (entidade competente). O MINUET emprega uma **gerência de coordenação e controle adaptativos e distribuídos** baseada no contexto temporal e espacial dos veículos (*crowdsourcing*), onde eles colaborativamente executam o monitoramento e a disseminação de maneira dinâmica e sob-demanda. Os principais diferenciais do MINUET são: (i) o controle distribuído, onde todos os dispositivos com capacidade de entregar as informações, o fazem; diferente dos atuais trabalhos, apenas o líder do grupo decide quais informações são enviadas a uma central de controle ou apenas o líder se comunica com a central; (ii) e o monitoramento colaborativo; diferente de trabalhos, onde apenas o líder monitora o evento, não havendo assim um monitoramento colaborativo. O MINUET foi avaliado através de simulações, e empregou-se o agrupamento de [Tal et al. 2016] para mensurar a eficácia do MINUET no gerenciamento da disseminação de eventos emergenciais. Os resultados mostram que o MINUET consegue detectar um evento cerca de seis vezes por minuto, monitorar e disseminar cerca de 40% do dados, através de agrupamentos, apoiando assim o tratamento destes eventos.

O restante do artigo está estruturado como descrito a seguir. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 descreve o sistema MINUET e o seu funcionamento. A Seção 4 detalha a metodologia de avaliação aplicada no MINUET e os resultados alcançados. Finalmente, a Seção 5 apresenta as considerações finais do trabalho.

2. Trabalhos Relacionados

A onipresença de veículos nas cidades foi uma das causas que impulsionou o desenvolvimento de trabalhos em VANETs. Existem várias soluções na literatura que utilizam VANETs para auxiliar no cotidiano das cidades ao tratar algum aspecto urbano específico. Dentre os vários serviços para cidades inteligentes, alguns utilizam estrutura de agrupamentos para apoiar o rastreamento de objetos-alvos e/ou a disseminação de informações.

Em [Huang et al. 2017] foi proposto um esquema de controle que utiliza algoritmo de agrupamento baseado em área de visão juntamente com técnicas adaptativas de *streams* de vídeo para auxiliar na vigilância das vias de trânsito. Grupos de veículos propagam informações visuais em tempo real. A solução busca escolher o líder de grupo

mais apropriado, o qual é o responsável por capturar imagens da via e repassar aos demais membros. Entretanto, apenas o líder do grupo captura as informações, não permitindo assim uma colaboração durante o monitoramento. Os autores em [Shi et al. 2017] propõem uma solução de agrupamento centralizada baseada em uma arquitetura de rede híbrida. Ao usar interfaces IEEE 802.11p e LTE, os veículos podem escolher o modo de comunicação e disseminação de mensagens de alerta para outros grupos da rede através de uma entidade central. Tais mensagens visam alertar os motoristas sobre colisões em cruzamentos e engarrafamentos. Contudo, nesta proposta apenas o líder do grupo se comunica com a rede infraestruturada, o que pode causar atrasos na disseminação das informações.

Em [Wang et al. 2018] os autores propõem um *framework* para disseminação de conteúdo em redes de acesso heterogêneas usando uma estratégia de gerenciamento de tráfego baseada em agrupamentos. O trabalho utiliza um sistema baseado em *crowdsensing* para coletar informações de eventos de forma colaborativa de forma a entregá-las a uma central de gerenciamento. Membros do grupo enviam mensagens referentes a um evento para o líder do grupo, onde este analisa os dados de forma a ter informações mais precisas sobre os eventos e decide pela transmissão imediata ou adiada para a central. Em [Khakpour et al. 2017] os autores apresentam e realizam testes em dois algoritmos de agrupamento para rastreamento de veículos em vias de trânsito. Na estratégia aplicada pelos algoritmos, os grupos são formados apenas pelos veículos que atualmente detectam o veículo alvo e por aqueles veículos que possuem uma alta probabilidade de detectar o veículo alvo logo em seguida. O líder do grupo concentra os dados coletados de todos os membros do grupo, analisa os dados de forma a eliminar redundâncias e os retransmite à rede infra-estruturada. Apesar da eficácia, estas duas soluções concentram no líder do grupo a responsabilidade de decidir o que enviar para uma central de controle.

No trabalho proposto em [Yedder and Benyahia 2017], os autores propõem um sistema reativo de disseminação de informação para auxiliar no despacho de veículos de emergência (e.g. uma ambulância). A partir de informações coletadas pelos veículos na VANETs, é possível inferir que uma via está congestionada, o que possibilita que o rota do veículo de emergência possa ser alterada em tempo real. Contudo, são feitas simulações apenas considerando questões de mobilidade e de tráfego, sem levar em conta questões da infraestrutura de rede (e.g. atraso, colisões) poderiam influenciar na solução. Em [Gorrieri et al. 2016] é proposta uma estratégia descentralizada para aquisição de dados de fenômenos em redes veiculares. Numa primeira fase (*downlink phase*), um nó sorvedouro remoto (pode estar na nuvem) dispara mensagens na rede para formação de grupos. Durante a formação, os líderes de cada grupo também já são definidos. Em uma segunda fase (*uplink phase*), dados coletados pelos veículos de cada grupo, referentes ao fenômeno observado, são enviados para os seus respectivos líderes e repassados para o nó sorvedouro. A ideia é criar uma topologia baseada em grupos para a disseminação de dados dos fenômenos. Contudo, esta solução emprega uma estratégia pró-ativa, o que pode desperdiçar recursos em casos onde não ocorrem fenômenos.

3. Sistema de Detecção e Disseminação de Eventos Emergenciais Urbanos

Esta seção apresenta o sistema MINUET para o apoio ao monitoramento cooperativo de eventos emergenciais urbanos através de agrupamentos sob demanda de veículos, que emprega uma gerência de coordenação e controle adaptativos e distribuídos baseada no contexto temporal e espacial dos veículos (*crowdsourcing*). Ele opera sobre VANETs

híbridas para apoiar o monitoramento de eventos emergenciais e a disseminação dos dados coletados para uma entidade externa que trata os eventos urbanos a partir do seu reconhecimento. Na entidade, os dados são analisados por aplicações alvo de modo a auxiliá-la na tomada de decisões à resolução dos eventos. O MINUET utiliza um controle distribuído, onde os próprios veículos decidem quem irá monitorar e/ou disseminar as informações. Inicialmente, descreve-se o modelo de ambiente urbano de atuação do MINUET. Em seguida, os componentes da arquitetura MINUET são detalhados e as suas interações, bem como é ilustrado o seu funcionamento.

3.1. Modelo do Ambiente Urbano

O ambiente urbano de atuação consiste de um espaço geográfico não estruturado formado por um conjunto de construções compartilhando espaço com malhas viárias onde veículos trafegam continuamente obedecendo legislações de tráfego, como ilustra a Figura 1. Neste ambiente, os veículos que trafegam são capazes de se comunicarem uns com os outros, Veículos-a-Veículos (V2V), e com a Infraestrutura Urbana através de Estações Bases (EB), Veículos-a-Infraestrutura (V2I), criando assim redes veiculares híbridas. Como ocorrem em zonas urbanas reais, eventos urbanos emergenciais de naturezas distintas podem surgir aleatoriamente no tempo e no espaço. Um evento urbano emergencial é qualquer evento que possa impactar no dia-a-dia da cidade e dos seus cidadãos, como por exemplo, incêndios, acidentes, delitos, obstruções nas vias, dentre outros. Estes eventos devem ser tratados por uma Entidade Externa (EE), representando uma autoridade competente, a qual se comunica, através das EBs, com a rede veicular não-estruturada.

O modelo de ambiente urbano assumido é composto por um conjunto V de n veículos (nós) identificados por $\{v_1, v_2, ..., v_n\}$ e um conjunto E de m eventos identificados por $\{e_1, e_2, ..., e_m\}$, onde qualquer veículo $v_i \in V$ pode detectar um evento $e_j \in E$, desde que e_j esteja no raio de alcance de v_i. Esta detecção acontece por meio de dispositivos de geolocalização e/ou sensores embarcados, e assume-se por raio de alcance a distância máxima do evento ao veículo de forma que o dispositivo embarcado consiga detectar o evento. Todo evento possui um período de existência, que vai desde o instante de sua ocorrência até o seu natural sumiço ou tratamento final (definido pela EE), e a sua posição no espaço pode variar ao longo do tempo, caracterizando assim um evento fixo ou móvel. Uma vez detectado, o evento precisa ser monitorado ao longo do tempo por um grupo de veículos, sendo que a formação de um grupo depende do evento ser detectado. Portanto, denota-se por $E' \subset E$, o conjunto de todos os eventos que foram detectados. Assim, para cada $ev \in E'$, definimos um grupo de veículos de monitoramento $Gm(ev) = \{vm\}$, onde $vm \subset V$. Então o modelo urbano ainda possui um conjunto G de r grupos de monitoramento identificados por $\{Gm_1, Gm_2, ..., Gm_r\}$. Vale ressaltar que o tamanho do grupo associado a um evento é dinâmico, ou seja, ele pode variar com o passar do tempo devido à mobilidade dos veículos e do próprio evento. Os veículos podem entrar e sair do grupo, conforme a densidade de veículos nas vias e de suas direções.

Na Figura 1, as elipses em vermelho representam os grupos de veículos de monitoramento responsáveis pelo monitoramento e disseminação da informações dos eventos existentes em um dado instante de tempo. A mobilidade do evento deve ser levada em conta tendo em vista que os grupos devem se manter próximos dos seus respectivos eventos. Logo, pode-se concluir que um grupo responsável por monitorar um evento móvel deve acompanhá-lo o maior tempo possível, enquanto que um grupo responsável por mo-

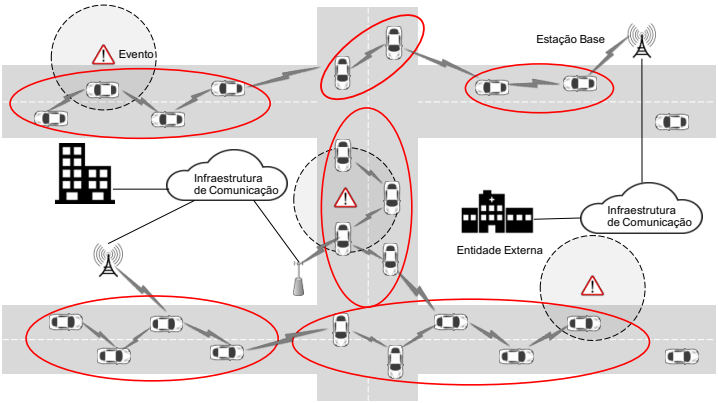


Figura 1. Modelo do ambiente urbano

nitorar um evento fixo tende a se afastar do respectivo evento (tomando o evento fixo como referencial). Assim, devido à alta dinamicidade da rede veicular e do eventos, intervalos de intermitência podem surgir resultando em possíveis lapsos no monitoramento.

Finalmente, para possibilitar um melhor tratamento do evento, os dados monitorados devem ser entregues à EE em um limite de tempo máximo de forma a serem passíveis de uso pelas suas respectivas aplicações alvo, e assim serem úteis à EE durante a tomada de decisão à resolução do evento. Obviamente, este limite de tempo depende diretamente da natureza do evento. Por exemplo, o limite de tempo para entregar os dados monitorados do evento *obstrução na via* pode ser maior do que aquele associado a um evento *veículo em fuga*. Define-se então o máximo de tempo de entrega $MTE(ev)$ como o intervalo de tempo máximo que o dado monitorado de $ev \in E'$ deve ser entregue ao destino.

3.2. MINUET

Para garantir o monitoramento cooperativo e dinâmico de eventos apoiado por agrupamentos é proposto o sistema MINUET (**M**onitorINg and Dissemination of **U**rban **E**ven**T**s). A arquitetura do MINUET é composta por dois módulos principais, chamados *Agrupamento* e *Controle*, como ilustrado na Figura 2. O módulo Agrupamento configura a formação e a manutenção dos grupos de veículos que irão monitorar o evento. O módulo Controle é responsável pela coordenação da detecção, anúncio e monitoramento dos eventos e pela disseminação das informações destes eventos às EEs.

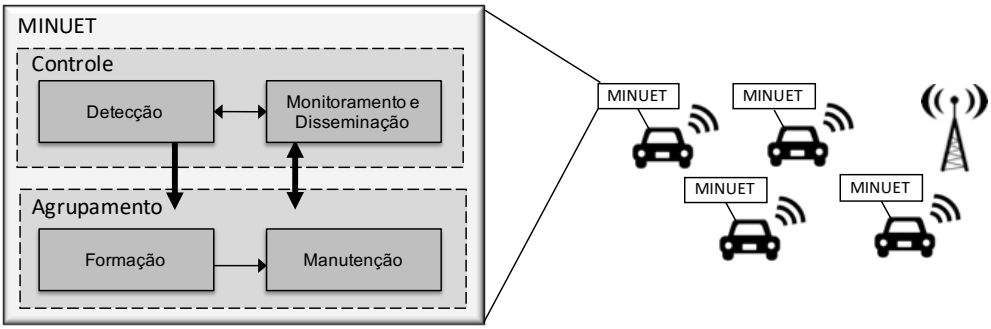


Figura 2. Arquitetura do MINUET

O módulo Agrupamento: tem por objetivo a criação e manutenção de grupos de veículos. Este módulo é composto por dois componentes: *Formação* e *Manutenção*.

O componente *Formação* é responsável pela descoberta da vizinhança, seleção do líder do grupo, definição dos papéis de cada nó no grupo e estabelecimento da comunicação dos nós com seus respectivos líderes. Inicialmente, um nó deve anunciar periodicamente sua existência para seus vizinhos enquanto que, simultaneamente, obtém informações semelhantes dos mesmos com o objetivo de construir a topologia de rede. Com base nas informações dos vizinhos, nós podem fazer ou não parte de um grupo. Aqueles que são escolhidos para fazer parte do grupo são denominados de nós membros. Além disso, um nó líder é selecionado para exercer funções de manutenção do grupo.

Uma vez formado o grupo, o componente de *Manutenção* é o responsável pelo reagrupamento e/ou refiliação de membros no grupo. Considerando fatores como a alta velocidade dos nós, mobilidade do evento, variação topológica da rede, comportamento dos condutores e interferências no canal, a frequência de entrada e saída de nós nos grupos pode variar consideravelmente com o tempo. Por isso é necessária a criação de mecanismos que consigam lidar com esta alta dinamicidade da rede sem comprometer o seu desempenho. Este componente também trata da sobreposição de grupos, quando dois ou mais grupos compartilham uma determinada região em um mesmo instante de tempo. Nestes casos, pode haver a necessidade de mesclar estes grupos sobrepostos em apenas um só. Salienta-se que diferentes técnicas de agrupamento podem ser usadas, variando assim os seus mecanismos de formação e manutenção de grupos, conforme cada caso.

O módulo Controle: é responsável por coordenar a detecção, o monitoramento do evento no grupo formado e a disseminação das informações para a EE. Ele é formado por dois componentes, sendo o primeiro responsável por detectar e anunciar o evento e o segundo componente por monitorar e disseminar as informações do evento detectado. Para isso, além da gestão do grupo em si, é necessário a gestão e o controle de monitoramento no grupo, onde veículos específicos devem se auto-escalonar dinamicamente para colaborar no tratamento de múltiplos eventos sobre diferentes tipos de cenários de tráfego. Assim, além dos papéis definidos pelo módulo de *Agrupamento*, este módulo também estabelece outros três papéis, que são: *(i) Monitorador*: enquanto um evento estiver no raio de alcance de um veículo, o mesmo será capaz de monitorá-lo. *(ii) Retransmissor*: veículo que apenas dissemina as informações dentro do grupo com o intuito de alcançar a EE. (iii) *Gateway*: veículo que entrega as informações do evento monitorado à EB e, consequentemente, à EE. O *gateway* também é o responsável por repassar as informações entre grupos. Um veículo também pode exercer mais de um papel simultaneamente. Por exemplo, um veículo pode ser tanto monitorador, porque está próximo do evento, como *gateway*, porque está ao alcance da EB.

O componente *Detecção* trata inicialmente da análise e detecção de um evento por um veículo. Um veículo *vd* na via de trânsito, pode detectar a ocorrência de um evento *ev* em um determinado instante de tempo. Após a detecção, *vd* coleta as dados de contexto de *ev*, que são *instante de tempo da detecção*, *localização*, *velocidade* e *direção*. Se o evento é fixo, então sua velocidade é zero. Por razões de simplificação, assumimos que a análise e detecção do evento são realizadas através de técnicas de processamento e análise de imagens obtidas por sensores de câmeras embarcadas nos veículos. Além disso, assumimos também que durante a análise do evento, *vd* consegue estimar o $MTE(ev)$.

Após a etapa de coleta das informações de contexto, inicia-se o processo para disseminá-las, através de mensagens de anúncio ao longo da rede. Contudo, nem todos

os veículos da malha viária urbana necessitam ter ciência da ocorrência do evento. Além disso, após um determinado tempo, o anúncio de um evento não é mais útil. Desta forma, pretende-se limitar a disseminação do anúncio a um determinado intervalo de tempo, reduzindo assim o *overhead* e possíveis perdas e atrasos na rede. Considerando que um veículo *vr* receba a mensagem de anúncio no instante tr e que o evento foi detectado no instante td, então *vr* repassa a mensagem se o tempo que ainda tem para retransmití-la ainda não ultrapassou o limite de tempo de entrega referente ao evento ($tr - td \leq MTE(ev)$). Caso contrário, a mensagem é descartada. Este limite temporal calculado por cada veículo que recebe a mensagem de anúncio delimita uma área chamada de Zona de Anúncio (ZA), a qual determina os veículos que podem fazer parte do grupo para colaborar no monitoramento do evento.

A Figura 3 ilustra um evento detectado e a ZA gerada com base neste evento, bem como os veículos que fazem parte da ZA. O evento é detectado pelos veículos *VD1* e *VD2*, e as mensagens de anúncio seguem as rotas estabelecidas pelas setas. Percebe-se que apesar do veículo *A* estar geograficamente mais perto do evento do que o veículo *B* ($dA < dB$), *B* faz parte da ZA, enquanto que *A* não. Isto pode acontecer quando a mensagem de anúncio demorar mais para chegar em *A* do que em *B*, o que pode fazer com que o MTE seja ultrapassado em *A*. Na Figura 3, a rota que a mensagem segue para chegar em *A* possui mais saltos do que a rota para *B*, ocasionando assim o atraso e o descarte da mensagem em *A*. Todos os outros veículos continuam disseminando a mensagem. Após um veículo receber uma mensagem de anúncio e confirmar que está na ZA, ele então verifica se fará parte do grupo que colaborará ao monitoramento do evento. Esta verificação é feita pelo módulo *Agrupamento*, de acordo com a técnica e o serviço de agrupamento desenvolvido e utilizado para apoiar a disseminação de informação.

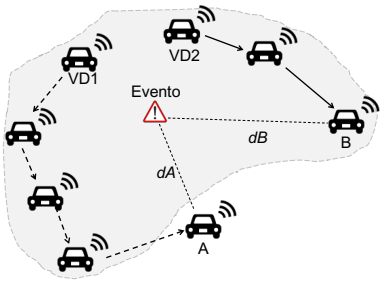


Figura 3. Zona de anúncio

Por fim, o componente de *Monitoramento e Disseminação* trata do monitoramento do evento e da disseminação de seus dados pelo grupo até entregá-las a uma EB. Para isto, este componente cuida do auto-escalonamento de papéis entre os veículos para o monitoramento cooperativo. Quando o veículo *vd* detecta o evento *ev*, ao mesmo tempo que realiza a disseminação do anúncio, também inicia o monitoramento de *ev*, coletando seus dados e disseminando através de mensagens de monitoramento. O veículo *vd* será então um *monitorador* enquanto *ev* estiver no seu raio de alcance. Caso *vd* também estiver ao alcance de uma EB, os dados coletados são entregues, fazendo com que *vd* também seja um *gateway*. Caso contrário, *vd* apenas dissemina as mensagens para cada vizinho. Cada veículo que recebe a mensagem de monitoramento, primeiro verifica se faz parte do grupo formado. Caso faça parte, também dissemina as mensagens, assumindo o papel de *retransmissor*. Se o veículo estiver ao alcance de uma EB, então ele assume o papel de

gateway, entregando assim as mensagens de monitoramento.

Algoritmo 1 Detecção do evento

Entrada: conjunto V de veículos, evento detectado ev

1: **para** cada veículo v_i em V **faça**
2: **se** ev está no alcance de v_i **então** ▷ v_i é *Monitorador*
3: Inicia agrupamento. Adiciona v_i a $Gm(ev)$ ▷ Solicita ao módulo Agrupamento
4: v_i cria e dissemina $msg.Anun(ev)$ e $msg.Mon(ev)$
5: **se** v_i está ao alcance de EB **então**
6: Entrega $msg.Mon(ev)$ a EB ▷ v_i é *Gateway*

O Algoritmo 1 descreve a operação que ocorre nos veículos que conseguem detectar o evento ev. Uma vez que ev é detectado, o veículo v_i assume o papel de *monitorador* e inicia a formação do grupo, que é realizado pelo módulo Agrupamento. Em seguida, v_i cria e dissemina as mensagens de anúncio e monitoramento. A mensagem de anúncio (*msg.Anun(ev)*) contém as informações de contexto do evento, necessárias para a manutenção do grupo formado, como por exemplo, a posição do evento, o tempo de detecção e o tipo do evento. A mensagem de monitoramento (*msg.Mon(ev)*) contém os dados monitorados do evento. Além disso, se v_i estiver ao alcance de uma EB, também assume o papel de *gateway* e entrega *msg.Mon(ev)* à EB.

Algoritmo 2 Disseminação do evento

Entrada: conjunto V de veículos, grupo de veículos $Gm(ev)$ que monitoram ev, msg do tipo $msg.Anun(ev)$ ou $msg.Mon(ev)$

1: **para** cada veículo v_i em V que recebe msg **faça**
2: Calcula $ZA(ev)$
3: **se** v_i está na $ZA(ev)$ **então** ▷ Senão estiver na $ZA(ev)$, descarta msg
4: **se** $msg == msgAnun(ev)$ **então** ▷ v_i recebeu mensagem de anúncio
5: Dissemina $msg.Anun(ev)$
6: Verifica se v_i pode ser adicionado a $Gm(ev)$ ▷ Solicita ao módulo Agrupamento
7: **senão** ▷ v_i recebeu mensagem de monitoramento
8: **se** $v_i \in Gm(ev)$ **então** ▷ Senão estiver em $Gm(ev)$, descarta msg
9: Dissemina $msg.Mon(ev)$ ▷ v_i é *Retransmissor*
10: **se** v_i está ao alcande de EB **então**
11: Entrega $msg.Mon(ev)$ a EB ▷ v_i é *Gateway*

Já nos veículos que não detectaram o evento e receberam uma mensagem de anúncio ou monitoramento, é necessário verificar se eles devem fazer parte do grupo de monitoramento e disseminar tais mensagens (Algoritmo 2). Toda vez que um veículo v_i recebe uma mensagem, ele deve verificar se está na ZA de ev. Se não estiver, descarta a mensagem. Caso esteja, se for uma mensagem de anúncio, ele dissemina a mensagem e verifica (através do módulo Agrupamento) se pode fazer parte do grupo $Gm(ev)$ que está monitorando ev. Se v_i receber uma mensagem de monitoramento, então só é possível disseminar se já fizer parte de $Gm(ev)$, fazendo neste caso que v_i seja um *retransmissor*. Além disso, se v_i estiver ao alcance de uma EB, também faz o papel de *gateway* e entrega a mensagem à EB. Caso não faça parte do grupo, descarta a mensagem.

A Figura 4 exemplifica um cenário de operação do MINUET em um intervalo de tempo *t* contendo o conjunto $V = \{A, B, C, D, E, F, G, H\}$ e o conjunto $E = \{EV\}$. Os veículos *D* e *E* conseguem detectar o evento *EV*, criando assim o conjunto $E' = \{EV\}$.

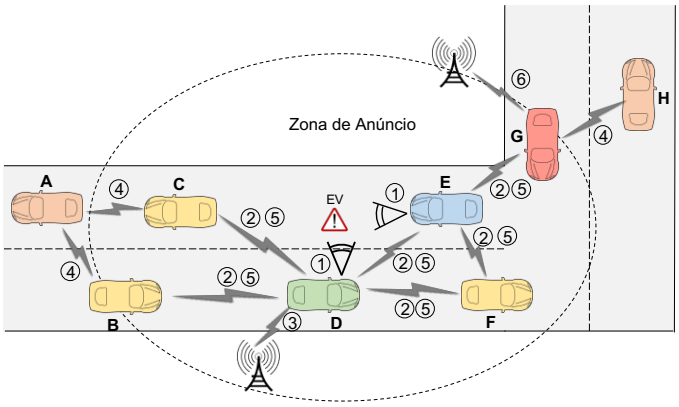


Figura 4. Exemplo de operação do MINUET

D e *E* coletam então os dados de contexto de *EV* (1), assumindo o papel de *monitoradores*. Imediatamente iniciam a disseminação das mensagens de anúncio (2). O veículo *D*, por estar ao alcance de uma EB, também entrega os dados monitorados (3), tornando-se assim também um *gateway*. Conforme as mensagens de anúncio vão sendo disseminadas, os veículos verificam se fazem parte da ZA e do grupo formado para realizar o monitoramento cooperativo. Os veículos *A* e *H*, ao receberem a mensagem de anúncio (4), verificam que não fazem parte da ZA, não participando assim do monitoramento de *EV*. Para efeitos de simplificação, consideramos que todos os veículos que estão na ZA também fazem parte do grupo. Assim, é criado então o grupo $Gm(EV) = \{B, C, D, E, F, G\}$. À medida que o grupo vai sendo formado, as mensagens de monitoramento também vão sendo disseminadas através dos seus membros (5). Os veículos *B*, *C* e *F* assumem então o papel de *retransmissores*, enquanto o veículo *G*, por conseguir entregar a mensagem de monitoramento à uma EB (6), assume o de *gateway*.

4. Avaliação

Esta seção apresenta uma avaliação do sistema MINUET para verificar a eficácia e eficiência no monitoramento e disseminação de eventos emergenciais urbanos. O MINUET foi implementado em C++ e testado no simulador NS3 juntamente com o SUMO. Foi utilizado um recorte do LuST (Luxembourg SUMO Traffic) como cenário de mobilidade de tráfego, onde os padrões de tráfego seguem os dados reais da cidade de Luxemburgo. O algoritmo empregado para realizar o agrupamento foi o DCA [Tal et al. 2016], ressaltando-se que outras técnicas de agrupamento podem ser utilizadas. Foram realizadas simulações em dois cenários distintos, ambos com um evento fixo e considerando diferentes velocidades média dos veículos, número de faixas na via de trânsito, sentido da via e distância da EB até o evento. Os dois cenários foram usados para avaliar como estas variações afetam no desempenho do MINUET. Em ambos os cenários ocorre apenas um evento com duração total de 10 mins.

A Figura 5 ilustra as vias onde ocorre a simulação, sendo que o centro das elipses indicam a posição onde ocorre o evento no cenário C1 e C2. Veículos que participam do grupo de monitoramento podem trafegar tanto na via onde o evento está localizado como em vias próximas. A Tabela 1 sumariza os parâmetros utilizados nas simulações dos dois cenários. As métricas utilizadas para aferir a eficácia e eficiência de MINUET foram: Número de veículos agrupados (Nva), Número de mensagens de monitoramento

entregues (Nme), Tempo de notificação do evento (Tne), Taxa de mensagens de monitoramento entregues (Tme), Taxa de mensagens de monitoramento entregues duplicadas (Tmd) e Número de detecções (Nd). A Tabela 2 descreve as métricas avaliadas.

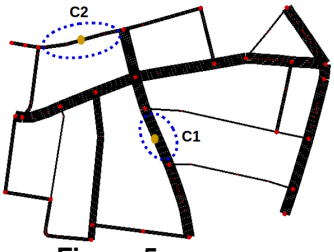


Figura 5. Cenário

Tabela 1. Configurações das simulações

Parâmetro	C1	C2
Tempo da simulação	12 mins	12 mins
Número de Eventos	1	1
Tempo de duração do evento	10 mins	10 mins
Velocidade média dos veículos	10,69 m/s	14,38 m/s
Sentido da via	mão dupla	mão única
Número de faixas	6	2
Distância do evento até a EB	130 m	160 m
Alcance de transmissão veículos	100 m	100 m

Tabela 2. Métricas aplicadas na avaliação

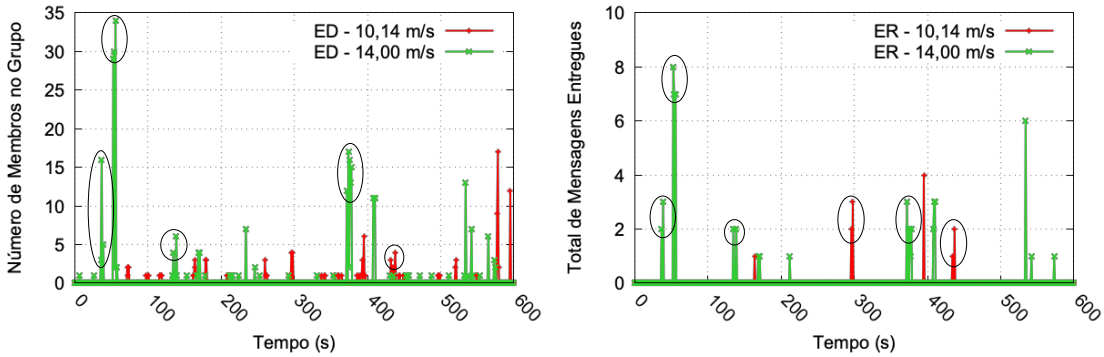
Métrica	Descrição
Nva	corresponde ao número de veículos participantes de um grupo
Nme	corresponde ao número total de mensagens de monitoramento que foram entregues a EB
Tne	corresponde ao intervalo de tempo entre o instante que o evento é detectado a primeira vez e o instante que a primeira mensagem de monitoramento é entregue à EB (Te)
Tme	corresponde ao percentual de mensagens de monitoramento enviadas pelos *monitoradores* que foram entregues à EB
Tmd	corresponde ao percentual de mensagens de monitoramento entregues a EB que são réplicas
Nd	corresponde ao número de vezes que o evento foi detectado durante a sua ocorrência

4.1. Resultados

A Figura 6 apresenta o desempenho do MINUET referente à detecção do evento e à entrega dos dados monitorados à EB. O gráfico 6(a) mostra o número de veículos por grupo (Nva) ao longo do tempo nos dois cenários, sendo que um grupo é formado a cada vez que o evento é detectado (*Evento Detectado* - ED). É possível perceber tanto momentos de intermitência do monitoramento, quanto momentos em que grupos formados mudam de tamanho ao longo do tempo, como destacados pelas elipses. Estas mudanças de tamanho são o resultado direto das funções de gerência do agrupamento, que controlam quando veículos entram e deixam o mesmo grupo. Isto demonstra o monitoramento cooperativo dos veículos alcançado pelo MINUET durante o monitoramento do evento (*crowdsourcing*). O gráfico também mostra que no cenário de 14 m/s, o número de veículos por grupo é maior que no cenário de 10,14 m/s. Isto se deve ao fato da densidade da via ser maior, logo há mais veículos trafegando. Também é possível notar que há uma grande quantidade de grupos com apenas um veículo. Isto é explicado porque o veículo *monitorador* está isolado, ou seja, não há nenhum outro veículo próximo o suficiente para participar do seu grupo de forma a entregar os dados monitorados a EB.

O gráfico 6(b) mostra a eficácia do MINUET ao entregar as mensagens de monitoramento à EB (*Evento Recebido* - ER). É possível observar que nem sempre que o evento é detectado, as mensagens de monitoramento são entregues. Isto acontece quando não há veículo algum do grupo próximo o suficiente da EB para entregar as mensagens. Além disso, ao comparar com o gráfico 6(a), as mensagens são entregues quando o Nva

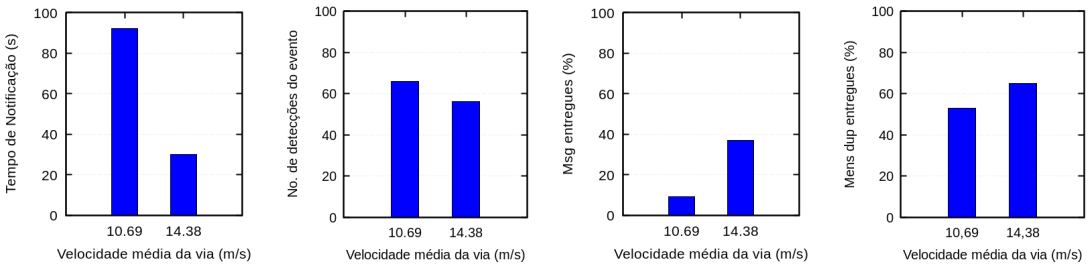
é maior, o que também faz sentido, uma vez que a probabilidade de haver um veículo ao alcance da EB é maior. Isto também explica o fato de haver mais mensagens entregues no cenário 2, já que o seu Nva médio é maior do que o Nva do cenário 1. Também pode-se observar que há mais mensagens sendo entregues (destacadas pelas elipses no gráfico) quando os grupos passam mais tempo formados. Ou seja, é possível inferir que a coordenação e a colaboração dentro do grupo formado é que permitem a entrega das mensagens a EB, demonstrando assim o controle distribuído alcançado pelo MINUET.



(a) Grupos formados (b) Mensagens entregues

Figura 6. Detecção do evento e entrega das mensagens de monitoramento

Os gráficos da Figura 7 apresentam o custo computacional do MINUET no monitoramento cooperativo. O gráfico 7(a) mostra o tempo decorrido entre o instante da primeira vez que o evento é detectado e o instante de notificação do evento (Tne), o que corresponde à primeira mensagem de monitoramento que chega na EB. Apesar de que no cenário 2 a velocidade média dos veículos é maior, o seu Tne é menor. Isto é explicado pelo fato de que no cenário 2 é formado um grupo com um Nva maior e mais cedo do que no cenário 1, possibilitando assim a entrega da mensagem de monitoramento. Já o gráfico 7(b) apresenta o número de vezes que o evento foi detectado ao longo de sua ocorrência (Nd). No cenário 1 o evento foi detectado mais de 60 vezes enquanto que no cenário 2 foi detectado cerca de 55 vezes. Apesar da intermitência de detecção, o sistema MINUET conseguiu alcançar uma média de 6 detecções por minuto.



(a) *Tne* (b) *Nd* (c) *Tme* (d) *Tmd*

Figura 7. Notificação evento, msgs entregues, msgs duplicadas e no. detecções

O gráfico 7(c) apresenta a percentual de mensagens de monitoramento não duplicadas entregues a EB (Tme). É possível ver que no cenário 2, o Tme chega perto de 40%, o que garante uma taxa razoável de dados coletados do evento, considerando o ambiente dinâmico onde ocorre. Finalmente, o gráfico 7(d) apresenta o percentual de réplicas de

mensagens de monitoramento entregues a EB (Tmd). Neste caso, os resultados mostram que nos dois cenários, mais de 50% das mensagens entregues são réplicas. Isto acontece pois pode haver mais de um *gateway* ao mesmo tempo em um grupo formado. Vale ressaltar que os resultados foram alcançados ao utilizar o algoritmo DCA. Extensões do trabalho estão em andamento utilizando outras técnicas de agrupamento, eventos móveis e sobrepostos no tempo e espaço para averiguar a escalabilidade da proposta.

5. Conclusões

Este trabalho apresentou o sistema MINUET para apoio ao monitoramento cooperativo de eventos emergenciais urbanos através de agrupamento sob demanda de veículos. O MINUET consegue tratar eventos aleatórios no tempo e espaço em um ambiente urbano através do monitoramento e disseminação cooperativos dos dados coletados até uma entidade externa. Uma vez na entidade externa, estes dados podem ser interpretados, gerando informações para auxiliar a entidade na tomada de decisão referente à resolução do evento. Resultados de simulações mostraram que o MINUET consegue detectar um evento, formar grupos que colaboram para a disseminação de mensagens de monitoramento e entregar as mensagens a uma EB através destes grupos. Estas mensagens podem ser usadas pela EE para auxiliar no processo de tomada de decisão referente à resolução do evento. Como trabalhos futuros pretende-se analisar o desempenho do MINUET em cenários com eventos sobrepostos no tempo e espaço, além de considerar eventos móveis e utilizar outras técnicas de agrupamento.

Referências

Abhishek, R., Zhao, S., and Medhi, D. (2016). Spartacus: Service priority adaptiveness for emergency traffic in smart cities using software-defined networking. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–4.

Castro, J., Araujo, I., Anjos, E., and Matos, F. (2017). A Survey on Bus Monitoring Systems. In *Computational Science and Its Applications – ICCSA 2017*, Lecture Notes in Computer Science, pages 220–231.

Chou, Y., Chu, T., Kuo, S., and Chen, C. (2017). An Adaptive Emergency Broadcast Strategy for Vehicular Ad Hoc Networks. *IEEE Sensors Journal*, 18(12):4814–4821.

Furutai, K. and Kanno, T. (2018). Resilience Analysis of Urban Critical Infrastructure: A human-Centred View of Resilience. Technical report, EPFL International Risk Governance Center.

Gorrieri, A., Martalò, M., Busanelli, S., and Ferrari, G. (2016). Clustering and sensing with decentralized detection in vehicular ad hoc networks. *Ad Hoc Networks*, 36:450 – 464. Vehicular Networking for Mobile Crowd Sensing.

Hassan, A. M. and Awad, A. I. (2018). Urban transition in the era of the internet of things: Social implications and privacy challenges. *IEEE Access*, 6:36428–36440.

Huang, C., Wang, H., Zhou, H., Xu, S., and Ren, D. (2017). EVAC-AV: The Live Road Surveillance Control Scheme Using an Effective-Vision-Area-Based Clustering Algorithm With the Adaptive Video-Streaming Technique. *IEEE Systems Journal*, 11(3):1228–1238.

Khakpour, S., Pazzi, R. W., and El-Khatib, K. (2013). A Distributed Clustering Algorithm for Target Tracking in Vehicular Ad-hoc Networks. In *Proceedings of the Third ACM*

International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications, DIVANet '13, pages 145–152, New York, NY, USA. ACM.

Khakpour, S., Pazzi, R. W., and El-Khatib, K. (2017). Using clustering for target tracking in vehicular ad hoc networks. *Vehicular Communications*, 9:83–96.

Ma, F., Liu, X., Liu, A., Zhao, M., Huang, C., and Wang, T. (2018). A Time and Location Correlation Incentive Scheme for Deep Data Gathering in Crowdsourcing Networks. *Wireless Communications and Mobile Computing*, 2018:22.

Macedo, R., Melo, R., Santos, A., and Nogueira, M. (2014). Experimental performance comparison of single-path and multipath routing in VANETs. In *GIIS*, pages 1–6.

Memos, V. A., Psannis, K. E., Ishibashi, Y., Kim, B.-G., and Gupta, B. (2018). An efficient algorithm for media-based surveillance system (eamsus) in iot smart city framework. *Future Generation Computer Systems*, 83:619 – 628.

Monajemi, S., Sanei, S., and Ong, S.-H. (2018). Information reliability in complex multitask networks. *Future Generation Computer Systems*, 83:485 – 495.

Monreal, C. O. (2018). Future urban mobility group: Smart cities research institute swinburne university of technology, melbourne australia [its research lab]. *IEEE Intelligent Transportation Systems Magazine*, 10(2):203–205.

Mousa, M., Zhang, X., and Claudel, C. (2016). Flash flood detection in urban cities using ultrasonic and infrared sensors. *IEEE Sensors Journal*, 16(19):7204–7216.

Quadros, C., Cerqueira, E., Santos, A. L. d., Lim, J.-H., and Gerla, M. (2015a). Beaconless video streaming management for VANETs based on QoE and link-quality. In *IEEE IM*, pages 191–198.

Quadros, C., Santos, A., Gerla, M., and Cerqueira, E. (2016). QoE-driven dissemination of real-time videos over vehicular networks. *Comp. Communications*, 91-92:133–147.

Quadros, C., Santos, A. L. d., Gerla, M., and Cerqueira, E. (2015b). A QoE-Aware Mechanism to Improve the Dissemination of Live Videos over VANETs. In *SBRC*, pages 31–40. IEEE Computer Society.

Shi, C., Zhou, Y., Li, W., Li, H., Lu, N., Cheng, N., and Yang, T. (2017). A centralized clustering based hybrid vehicular networking architecture for safety data delivery. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6.

Silva, C., Nogueira, M., Kim, D., Cerqueira, E., and Santos, A. (2016). Cognitive radio based connectivity management for resilient end-to-end communications in VANETs. *Computer Communications*, 79:1–8.

Tal, I., Kelly, P., and Muntean, G. (2016). A novel direction-based clustering algorithm for vanets. In *23rd International Conference on Telecommunications (ICT)*, pages 1–5.

Timashev, S. A. (2017). Resilient urban infrastructures – basics of smart sustainable cities. *IOP Conference Series: Materials Science and Engineering*, 262(1):012197.

Wang, X. (2018). Data Acquisition in Vehicular Ad Hoc Networks. *Commun. ACM*, 61(5):83–88.

Wang, X., Ning, Z., Hu, X., Wang, L., Hu, B., Cheng, J., and Leung, V. C. M. (2018). Optimizing content dissemination for real-time traffic management in large-scale internet of vehicle systems. *IEEE Transactions on Vehicular Technology*, pages 1–1.

Yedder, H. B. and Benyahia, I. (2017). Reactive emergency vehicles dispatching based real-time information dissemination. In *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 471–477.

Gerenciamento da Infraestrutura de Comunicação baseado na Distribuição Gaussiana do Tempo de Contato entre Veículos e a Infraestrutura

Lucas D. Silva[1], Fernanda S. H. Sumika[3], Daniel L. Guidoni[3], João F. M. Sarubbi[4], Cristiano G. Pitangui[2], Cristiano M. Silva[1,2]

[1]Universidade Federal de Ouro Preto (DECOM/UFOP) – MG – Brasil

[2]Universidade Federal de São João del-Rei (DTECH/UFSJ) – MG – Brasil

[3]Universidade Federal de São João del Rei (DCOMP/UFSJ) – MG – Brasil

[4]Centro Federal de Educação Tecnológica (DECOM/CEFET) – MG – Brasil

lucas.diniz@aluno.ufop.edu.br, joao@decom.cefetmg.br

{fsumika, guidoni, pitangui.cristiano, cristiano}@ufsj.edu.br

Abstract. *Seeking novel solutions for urban mobility seems paramount for 21st century society. This work discusses the performance of vehicular networks relying on the Delta Network. Delta characterizes the connectivity of the vehicular network by the percentage of travel time in which the vehicles are connected to the infrastructure. This work presents a novel deployment strategy based on the Gaussian distribution of the percentage of the trip duration in which vehicles are covered by the infrastructure. Roadside units are distributed throughout the urban area through a heuristic based on weights assigned according to the trips that are covered by the new roadside unit. The most important gain of this approach is that the resulting coverage pattern favors most of the fleet experiencing similar connectivity, while vehicles with very low or very high connectivity become the exception. Results show that the strategy promotes the distribution of contact time according to the Gaussian distribution.*

Resumo. *Buscar novas soluções para a mobilidade urbana torna-se primordial para a sociedade do século XXI. Este trabalho discute o desempenho de redes veiculares através da Rede Delta. Delta caracteriza a conectividade da rede veicular pela porcentagem de tempo de viagem em que os veículos estão conectados à infraestrutura. Este trabalho apresenta uma estratégia de deposição para redes veiculares baseada na distribuição gaussiana do percentual da viagem em que veículos estão cobertos pela infraestrutura. Unidades de comunicação são distribuídas pela área urbana através de uma heurística baseada em pesos atribuídos de acordo com as viagens que são cobertas pela nova unidade. Como principal benefício dessa abordagem, o padrão de cobertura resultante favorece que a maior parte da frota experimente conectividade semelhante, enquanto que veículos com baixíssima ou altíssima conectividade se tornam exceção. Os resultados indicam que a estratégia promove a distribuição de tempos percentuais de contato de veículos com a infraestrutura em acordo com a distribuição gaussiana.*

1. Introdução

Nas últimas décadas, o crescimento da população urbana tem sido notável. Atualmente, 50% da população vive em áreas urbanas, e previsões indicam que até 2050 esse número chegará a 70% [Prospects, 2007]. Hoje, 64% de todas as viagens feitas estão dentro de ambientes urbanos e a quantidade total de quilômetros urbanos percorrida deverá triplicar até 2050 [Van Audenhove et al., 2014]. Assim, buscar novas soluções para a mobilidade urbana torna-se primordial para a sociedade do século XXI. Veículos autônomos [Lozano-Perez, 2012] e compartilhamento de veículos [Katzev, 2003] tendem a assumir cada vez mais os deslocamentos diários nos centros urbanos. Gradualmente, carros tendem a assumir o papel de serviços contratados, ao invés de bens de consumo, moldando uma nova indústria automobilística, influenciando a forma como são projetados, produzidos e comercializados.

A tecnologia de comunicação embarcada em veículos assumirá, cada vez mais, caráter essencial. Sistemas de navegação e automação do veículo podem se integrar à via para aumento da eficiência energética, promovendo a regulação do tráfego, mitigação de colisões e sincronização de viagens, fazendo com que a frota em operação seja minimizada em relação à demanda por deslocamentos. A integração digital dos veículos à infraestrutura viária e às demais entidades do sistema de trânsito numa rede única de comunicação permitirá, em fato, que a própria infraestrutura se reorganize. Vias inteligentes podem alterar sua velocidade permitida, barrar a permissão para estacionar, mudar a programação semafórica [Silva et al., 2015a] e, até mesmo, alterar seu sentido de tráfego, implementando corredores dinâmicos de breve duração para escoar rapidamente excedentes de tráfego que poderiam acarretar congestionamentos.

Funcionalidades com esse nível de criticidade dependem do projeto e disponibilização de redes de comunicação específicas e geridas por autoridades governamentais, com acesso controlado e altos níveis de confiabilidade e segurança. Diante disso, este trabalho aprofunda nosso estudo sobre estratégias para o projeto de redes veiculares infraestruturadas apresentando uma heurística baseada em pontos (*scores*) para definir os locais onde as unidades de comunicação (antenas) veicular devem ser implantadas numa cidade, de forma a se garantir metas pré-definidas de desempenho da rede. O desempenho é mensurado a partir da Rede Delta [Silva and Meira Jr., 2015], que caracteriza a conectividade da rede veicular pela porcentagem de tempo de viagem em que os veículos estão conectados à infraestrutura depositada ao longo da malha viária. A heurística considera uma estratégia baseada na distribuição gaussiana dos tempos de contatos de veículos com a infraestrutura.

Como principal benefício dessa abordagem, o padrão de cobertura resultante favorece que a maior parte da frota experimente conectividade semelhante, enquanto que veículos com baixíssima ou altíssima conectividade se tornam exceção. Os experimentos consideram o rastro de mobilidade realístico da cidade de Colônia [Uppoor et al., 2014]. Os resultados demonstram que a heurística atinge seu objetivo, promovendo a distribuição de tempos percentuais de contato de veículos com a infraestrutura em acordo com a distribuição gaussiana. A estratégia é comparada com a aplicação da mesma heurística usando uma distribuição de probabilidades uniforme, bem como a deposição da infraestrutura nos locais de maior tráfego.

Este trabalho encontra-se organizado da seguinte forma: A seção 2 apresenta os

trabalhos relacionados. A seção 3 revisa a Rede Delta. A seção 4 apresenta a estratégia **gaussiana**. A seção 5 apresenta as estratégias usadas como *baseline*. A seção 6 apresenta os experimentos realizados. A seção 7 conclui o trabalho.

2. Trabalhos Relacionados

A literatura apresenta várias estratégias para o planejamento da infraestrutura de comunicação para redes veiculares. Uma descrição detalhada desses esforços é apresentada por Silva et al. [2017]. A probabilidade de contato é considerada nos seguintes trabalhos. Zheng et al. [2010] apresentam a avaliação da estratégia de implantação considerando a oportunidade de contato. A métrica mensura a fração da distância (ou tempo) que o veículo está em contato com a infraestrutura, enquanto que Lee and Kim [2010] propõem uma heurística gulosa para alocar a infraestrutura com o objetivo de melhorar a conectividade entre veículos, ao mesmo tempo em que reduzem suas desconexões, e cada interseção é considerada como uma potencial posição para uma unidade de comunicação. Pereira et al. [2018] apresentam uma modelagem multi-objetivo do planejamento de infraestrutura para a distribuição de *streaming* em redes veiculares.

Xie et al. [2013] abordam o problema de alocação de unidades de comunicação assumindo conhecimento prévio da trajetória de cada veículo. Baseando-se em dados históricos, os autores propõem um modelo probabilístico para inferir as localizações das unidades de comunicação. Já, Chi et al. [2013] consideram uma estratégia que distribui as unidades de comunicação tão equilibradamente quanto possível. No entanto, os autores comparam seus resultados com uma estratégia de alocação aleatória, que incorre em um desempenho muito baixo. De forma complementar, Liu et al. [2013] propõem uma estratégia de alocação de unidades de comunicação para a transferência de arquivos em redes veiculares modelando o encontro entre veículos e unidades de comunicação como uma cadeia da Markov homogênea de tempo contínuo.

Trullols et al. [2010] e Cataldi and Harri [2011] propõem a modelagem de alocação de unidades de comunicação como um problema de máxima cobertura, assumindo-se a premissa de conhecimento prévio das trajetórias dos veículos. Essa premissa é relaxada nos trabalhos [Silva et al., 2014, 2015b, 2016], que propõem modelos de complexidades crescentes usando apenas a taxa de migração de veículos entre regiões. Estudos analíticos também são encontrados na literatura. Bazzi et al. [2011] abordam os sistemas celulares como a solução mais viável a curto prazo para coletar mensagens informativas de veículos para um centro de controle remoto. O trabalho propõe um modelo matemático para avaliar o impacto do serviço imaginado na capacidade e cobertura de sistemas celulares em cenários simplificados. Os resultados mostram que a aquisição de pacotes pequenos e freqüentes de veículos é afetada por interferências mais do que outros serviços, como o serviço de voz.

Finalmente, o trabalho apresentado por Zheng et al. [2009] serve como inspiração para a Rede Delta [Silva and Meira Jr., 2015] revisada na seção 3. Os autores propõem a Cobertura Alpha para fornecer garantias no intervalo de interconexão medido em termos de distância percorrida. Uma rede veicular é considerada de cobertura α se um veículo percorrendo qualquer caminho de comprimento α passa, ao menos, por uma unidade de comunicação infraestruturada. Já a Rede Delta [Silva and Meira Jr., 2015] considera tempo de deslocamento, ao invés de distância percorrida.

Diferentemente das abordagens anteriores, a estratégia aqui proposta considera a implantação de unidades de comunicação (antenas) de forma a realizar uma distribuição gaussiana dos tempos em que veículos permanecem conectados à infraestrutura.

3. Avaliando a Conectividade Veicular Através da Métrica Rede Delta

A Rede Delta Silva and Meira Jr. [2015] avalia o tempo de conexão dos veículos em relação à duração viagem, caracterizando uma dimensão do desempenho da rede veicular. Sua representação é dada por $\Delta_{\rho_2}^{\rho_1}$, onde o parâmetro ρ_1 representa a porcentagem de tempo de viagem conectado, enquanto que ρ_2 indica porcentagem de veículos que experimentam a conectividade indicada por ρ_1. Sendo assim, uma rede veicular torna-se $\Delta_{\rho_2}^{\rho_1}$ quando ρ_2 por cento dos veículos experimentam conexão com unidades de comunicação (antenas) durante ρ_1 por cento da viagem.

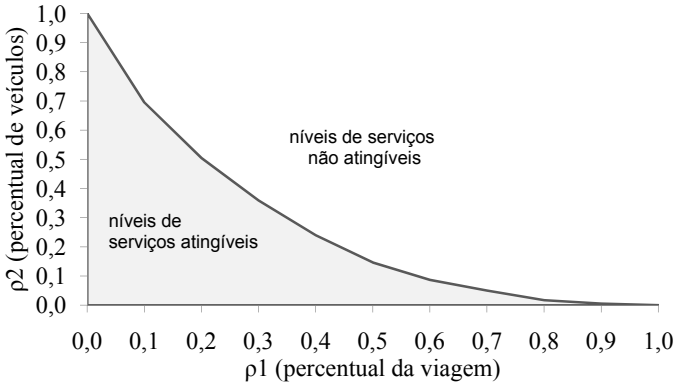
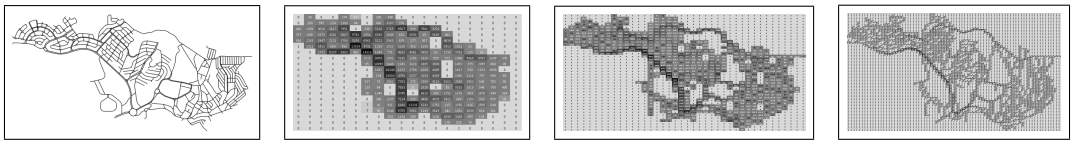


Figura 1. Figura apresenta a Rede Delta que representa a relação entre ρ_1 (percentual de duração da viagem sob cobertura) versus ρ_2 (percentual de veículos que recebem tal cobertura). Fonte: [Silva and Meira Jr., 2015].

A figura 1 exemplifica a métrica Delta. O eixo x representa ρ_1 (porcentagem da viagem conectado), enquanto que o eixo y indica ρ_2 (porcentagem de veículos conectados). Assim, Delta indica a relação entre ρ_1 e ρ_2 para toda a frota que se desloca numa dada região. A área abaixo da curva representa os níveis de conectividade atingíveis pela rede. Fora da curva, os níveis de conectividade não podem ser alcançados. Como a Rede Delta descreve uma relação de conectividade, ela pode ser utilizada para o planejamento e expansão de redes veiculares de forma a se atingir níveis pré-definidos de conectividade, uma aplicação fundamental para se garantir que a rede suporte as aplicações para as quais foi projetada.

4. Estratégia Gaussiana: Promovendo uma Distribuição Natural do Percentual de Tempo Viajado em que Veículos estão Cobertos

Estratégias de solução locais para a Rede Delta já foram apresentadas nos trabalhos [Sarubbi and Silva, 2016; Silva and Meira Jr, 2015]. No entanto, essas soluções focam na descoberta dos locais que precisam receber cobertura de forma a se obter a configuração de unidades de comunicação (antenas) para uma única tupla $\{\rho_1,\rho_2\}$, fazendo com que os demais pontos da curva Delta surjam como consequência da descoberta desse ponto inicial. No entanto, encontrar soluções para um único par $\{\rho_1, \rho_2\}$ pode não garantir boa eficiência para outras combinações de $\{\rho_1,\rho_2\}$.



(a) Malha Viária. (b) *Grade* 20×20. (c) *Grade* 40×40. (d) *Grade* 80×80.

Figura 2. Particionamento da malha viária em células urbanas.

Assim, essa investigação é expandida para se obter todos os pontos da curva. Para tanto, considera-se uma estratégia baseada em pontuações (*scores*) atribuídas para áreas da malha viária. Em fato, malhas viárias podem assumir topologias arbitrárias que resultam da forma de ocupação, relevo, dentre outros aspectos. Esse trabalho propõe uma estrutura baseada em grade (*grid*), conforme mostrado na figura 2, onde a cidade é dividida em células adjacentes de mesmo tamanho. Este tamanho pode variar de acordo com a precisão e cobertura esperada. A escolha das células que receberá cobertura se dá a partir da heurística geral abaixo apresentada em [Silva et al., 2018], que considera soluções de uma perspectiva global para toda a curva Delta.

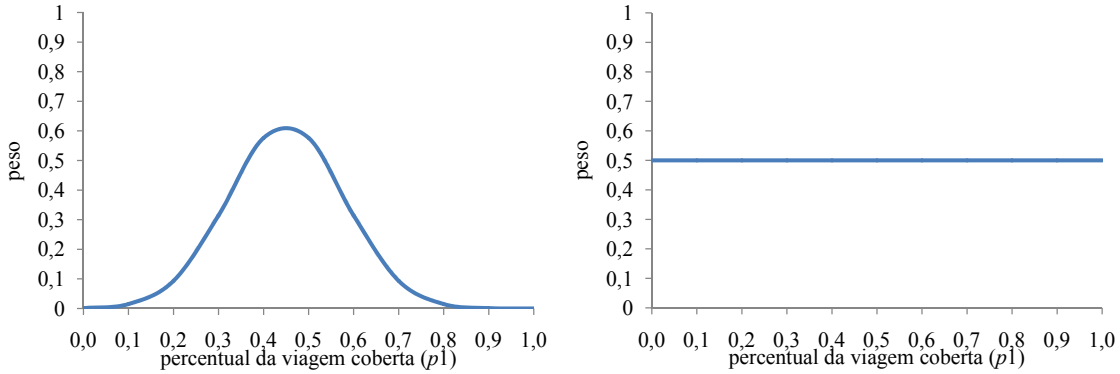
Algoritmo 1 Heurística geral

entrada: T, α; ▷ recebe trajetórias de veículos e número de células que devem ser cobertas.
saída: células_cobertas ▷ α células cobertas.
1: células_cobertas $\leftarrow \emptyset$; ▷ reseta conjunto contendo células cobertas
2: **enquanto** |células_cobertas| $< \alpha$ **faça** ▷ repete até cobrir α células
3: **para toda** célula descoberta $[x][y]$ no gride **faça**
4: **adiciona_célula**$[x][y]$ em células_cobertas; ▷ cobre temporariamente a célula
5: pontuação$[x][y] \leftarrow$ **computa_pontuação** depois de adicionar célula $[x][y]$;
6: **remove_célula** $[x][y]$ de células_cobertas; ▷ descobre a célula
7: **fim para**
8: $[x'][y'] \leftarrow$ **max_pontuação**(pontuação[][]); ▷ obtém a célula com maior pontuação
9: **adiciona_célula**$[x'][y']$ em células_cobertas; ▷ cobre definitavemente a célula
10: **fim enquanto**
11: **retorne** células_cobertas;

O algoritmo 1 apresenta a heurística gulosa geral que recebe como entrada (i) o rastro de veículos (T) e (ii) o número de células disponíveis a serem cobertas. O algoritmo inicia com um conjunto vazio contendo as células a serem cobertas (linha 1). Como nenhuma célula é coberta inicialmente, a heurística começa cobrindo temporariamente a célula [0,0] e avalia o desempenho da rede em termos de $\{\rho_1,\rho_2\}$. Em seguida, ela remove a cobertura de [0,0] e move para [0,1] e recalcula o desempenho da rede. Esse procedimento segue até avaliar todos os locais do gride (linhas 3-7). Em seguida, seleciona a célula que apresenta a pontuação mais alta. A célula que apresenta a pontuação mais alta é selecionada permanentemente para receber cobertura (linhas 8-9). Então, a heurística faz um laço até selecionar α células cobertas.

O cálculo de pontuação (linha 5) merece atenção especial, pois permite personalizar a forma de seleção das células, determinando o padrão de cobertura obtido. Tem-se a oportunidade de impor propriedades específicas na rede veicular, como selecionar o tipo de viagem que pretendemos priorizar e como a conectividade é

distribuída entre veículos. O estudo aqui apresentado considera pesos atribuídos com base numa distribuição normal padrão sobreposta ao eixo x da figura 1, isto é, ρ_1 que representa o percentual de tempo viajado. A distribuição normal padrão é centrada no ponto ρ_1=0.5, atribuindo-se maior pontuação para células que colaboram para que veículos estejam cobertos durante 50% do tempo viajado. Assim, esse trabalho avalia uma nova estratégia para a alocação de pesos baseada na distribuição padrão normal. A personalização da cobertura se dá através da valoração diferenciada de cada região do eixo x do gráfico da Rede Delta. Dito de outra forma, personaliza-se o perfil de cobertura que a rede deve ofertar.



(a) Estratégia Gaussiana (proposta) (b) Estratégia Balanceada (baseline)

Figura 3. Distribuição de pesos para implantação de unidades de comunicação, de acordo com o percentual da viagem coberto. A figura (a) apresenta a estratégia proposta, onde a pontuação máxima é atribuída para células que contribuem para viagens cobertas durante 50% do percurso, tipo de cobertura que será priorizada pela heurística. A figura (b) apresenta a distribuição de pesos considerada como *baseline*, onde todas as viagens apresentam igual prioridade.

A figura 3(a) apresenta a distribuição de pesos considerada. O eixo x indica a duração da viagem. O eixo y indica o peso considerado. Essa curva é uma distribuição normal padrão com média deslocada para o ponto x=0.5. Já a figura 3(b) apresenta uma distribuição de pesos uniformemente distribuída ao longo do eixo x. Essa é a distribuição de pesos adotada pela estratégia **balanceada**, usada como *baseline*. A distribuição uniforme atribui pesos idênticos para todas as viagens, não havendo priorização. Como pode ser percebido pela figura 3(a), a estratégia **gaussiana** atribui pesos maiores para valores localizados nas vizinhanças de ρ_1=0.5. Lembre-se que ρ_1 indica "duração da viagem sob cobertura". Assim, a atribuição de pesos **gaussiana** acaba por incentivar a cobertura de áreas que resultem em mais viagens atingindo cobertura durante 50% do tempo viajado do veículo.

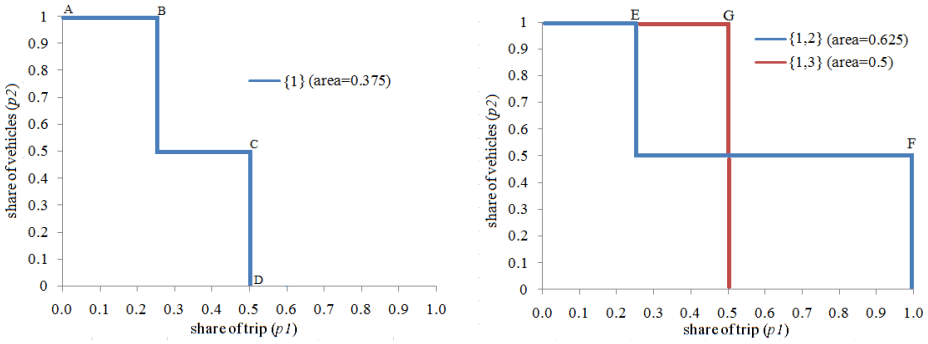
5. Baseline

São consideradas duas estratégias para se avaliar o desempenho da heurística proposta. A **estratégia intuitiva** cobre as células que possuem maior tráfego de veículos até que se atinja os níveis desejados de desempenho da rede. Já, a **estratégia balanceada** considera a mesma heurística geral apresentada no algoritmo 1 conjugada com a distribuição uniforme de pesos apresentada na figura 3(b), onde todos os tipos de viagem recebem a

mesma priorização. A estratégia balanceada é descrita em detalhes em [Silva et al., 2018], sendo a maximização da área sob a curva Delta sua característica mais interessante.

5.1. Estratégia balanceada: distribuição uniforme de pesos, independentemente do percentual de duração da viagem sob cobertura: uma alternativa para maximizar a área sob a curva Delta

Suponha apenas dois veículos viajando ao longo da malha viária dividida em 5 células. Suponha, também, que tais veículos permaneçam a mesma quantidade de tempo dentro de cada célula, e que existam antenas capazes de cobrir apenas duas células (α=2). O veículo v_1 cruza as células {1,2}, enquanto que o veículo v_2 cruza as células {1,3,4,5}.



(a) 1a iteração da estratégia **balanceada**. (b) 2a iteração da estratégia **balanceada**.

Figura 4. Exemplo didático sobre como a estratégia balanceada funciona. A figura (a) apresenta a Rede Delta depois de cobrir a *célula 1*. A figura (b) apresenta a Rede Delta considerando duas opções: a) cobrindo *célula 2*; b) cobrindo *célula 3*. Como cobrir a *célula 2* leva a uma área maior sob a curva, a *célula 2* é selecionada. Fonte: Silva et al. [2018].

$$v_1=\{1,2\};$$
$$v_2=\{1,3,4,5\};$$

Como a *célula 1* é cruzada por ambos os veículos, ela intuitivamente maximiza a área sob a curva Delta. Depois de cobrir a *célula 1*, têm-se o veículo v_1 coberto durante 50% da viagem e o veículo v_2 coberto durante 25% da viagem. A figura 4(a) plota a Rede Delta para este cenário hipotético. O ponto A é (ρ_1=0,00, ρ_2=1,00), ou seja, 100% dos veículos são cobertos durante 0% da duração da viagem (veículos v_1 e v_2). O ponto B é (ρ_1=0,25, ρ_2=1,00), ou seja, 100% dos veículos são cobertos durante 25% da duração da viagem (veículos v_1 e v_2). O ponto C é (ρ_1=0.50, ρ_2=0.50), ou seja, 50% dos veículos são cobertos durante 50% da duração da viagem (veículo v_1). O ponto D é (ρ_1=0,50, ρ_2=0,00), ou seja, nenhum veículo é coberto por mais de 50% da duração da viagem.

5.2. Estratégia Intuitiva

Já a estratégia **intuitiva** cobrir locais seguindo a ordem de popularidade. A célula mais popular é coberta primeiro. Recebe como entrada o volume de tráfego por célula (V), e o número de células a serem selecionadas (α), conforme apresentado no algoritmo 2.

Algoritmo 2 Estratégia intuitiva.

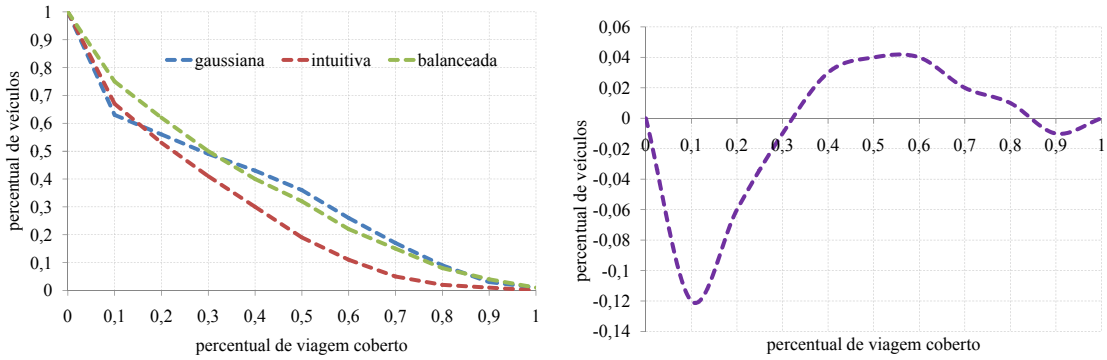
entrada: V, α;
saída: Υ; ▷ quantidade de células a serem cobertas
1: $V' \leftarrow$ **ordene**(V); ▷ ordene células conforme volume de tráfego
2: $\Upsilon \leftarrow$ **selecione_primeiros_elementos**(V',α); ▷ obtenha os primeiro α elementos de V'
3: **retorne** Υ;

6. Avaliação Experimental

Os experimentos são conduzidos usando-se o rastro de mobilidade veicular da cidade de Colônia, Alemanha [Uppoor et al., 2014][1] composto por 10 mil segundos de tráfego e mais de 75 mil veículos. A malha viária foi particionada num gride de 100×100 células urbanas, e avalia-se o desempenho da rede considerando a cobertura de [5%, 10%, 15% e 20%] da malha viária que possui tráfego.

6.1. Avaliando a QoS provida pela rede

Nesse estudo, busca-se avaliar a QoS provida pela rede a partir da deposição de unidades de comunicação proposta, comparada às duas estratégias de *baseline*. Cada estratégia deposita unidades de comunicação de forma cobrir 5%, 10% ou 15% da área urbana. A figura 5(a) plota a QoS apurada em termos da Rede Delta (percentual da viagem em que veículo viaja com cobertura (eixo x) *versus* quantidade de veículos (eixo y). A curva azul apresenta os resultados da estratégia **gaussiana** (proposta). Nota-se que essa estratégia oferece uma quantidade similar de veículos experimentando cobertura quando comparada à estratégia **intuitiva** (curva vermelha) para $\rho_1 < 0,2$ (eixo x da figura 5(a)).

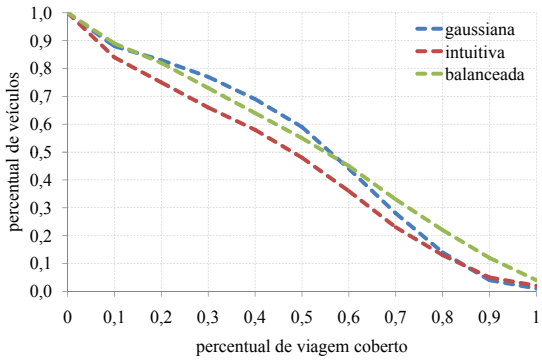


(a) cobertura de 5% da malha viária. (b) diferença entre **gaussiana** e **balanceada** (5%).

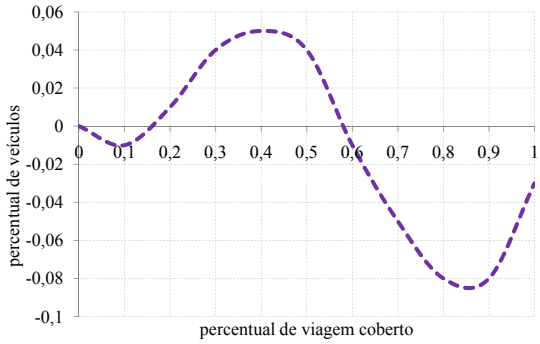
Figura 5. Cenário de cobertura de 5% da malha viária. A figura (a) apresenta a Rede Delta para cada uma das três estratégias: a) gaussiana (cor azul), b) intuitiva (cor vermelha), a) balanceada (cor verde). O eixo x indica o percentual da viagem coberto (ρ_1), enquanto que o eixo y indica o percentual de veículos (ρ_2). A figura (b) plota a diferença gaussiana menos balanceada.

Nesse intervalo, a estratégia **balanceada** (cor verde) vence as demais. No entanto, quanto mais próximo da metade do eixo x, melhor é o resultado apresentado pela estratégia **gaussiana**. Isso deve-se ao esquema de pontuação gaussiano proposto que valoriza a cobertura de células que resultem em viagens com cobertura de 50% do tempo

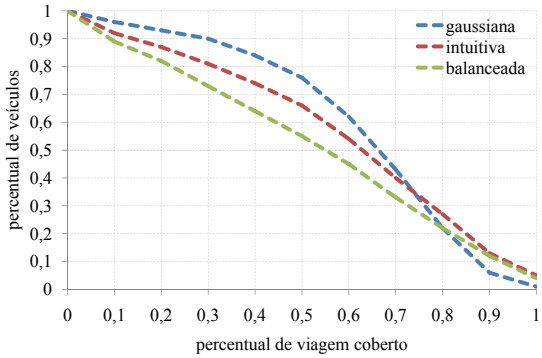
[1]Disponível em: `http://kolntrace.project.citi-lab.fr/`.



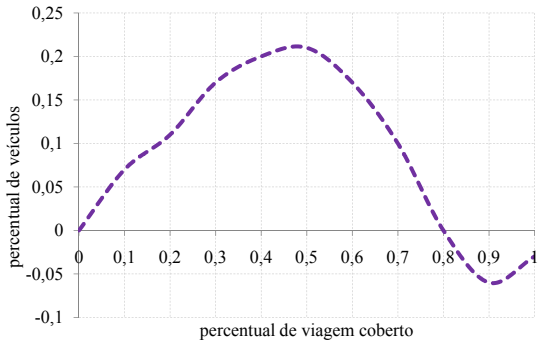
(a) cobertura de 10% da malha viária.



(b) diferença entre **gaussiana** e **balanceada** (10%).



(c) cobertura de 15% da malha viária.



(d) diferença entre **gaussiana** e **balanceada** (15%).

Figura 6. As figuras (a) e (b) apresentam o cenário de cobertura de 10% da malha viária, enquanto que as figuras (c) e (d) apresentam a cobertura de 15% da malha. A estratégia gaussiana é representada na cor azul, intuitiva na cor vermelha e balanceada na cor verde. O eixo x indica o percentual da viagem coberto (ρ_1), enquanto que o eixo y indica o percentual de veículos (ρ_2).

de deslocamento. Por outro lado, a estratégia **balanceada** maximiza a área abaixo da curva Delta [Silva et al., 2018]. Já a figura 5(b) plota apenas a diferença de cada ponto da estratégia **gaussiana** subtraído do mesmo ponto na estratégia **balanceada**. Curva acima do eixo x indica que a estratégia **gaussiana** apresentou mais veículos atendidos. Ao analisar o gráfico, percebe-se que a estratégia **gaussiana** apresenta mais veículos experimentando cobertura de 30% até 85% da viagem (veja o eixo x da figura 5(b) entre os valores 0.3 e 0.8). Note que o pico da diferença é atingido exatamente em x=0.5.

As figuras 6(a) e 6(b) apresentam o cenário onde 10% da malha viária é coberto. Nota-se que a estratégia **intuitiva** apresenta menor quantidade de veículos para, praticamente, toda a curva. A estratégia **gaussiana** se destaca no intervalo $0.2 \leq x \leq 0.6$. Para $x > 0.6$, a estratégia **balanceada** apresenta melhor desempenho. A visão comparativa (**gaussiana** menos **balanceada**) é apresentada na figura 6(b). O fato de não se notar um pico no ponto x=0.5 na figura 6(b) deve-se ao fato da estratégia **balanceada** ter obtido um bom volume de veículos cobertos nessa faixa em decorrência da maximização da área sob a curva Delta, obra do acaso, visto que a estratégia não possui como objetivo tal comportamento. Finalmente, as figuras 6(c) e 6(d) apresentam o cenário de cobertura de 15% da malha viária, onde nota-se, novamente maior volume de veículos cobertos pela estratégia **gaussiana** quando x=0.5 na figura 6(d).

6.2. Layout de Cobertura

Essa seção apresenta o *layout* de disposição das células cobertas. A figura 7(a) mostra o tráfego na cidade de Colônia, Alemanha. A intensidade da cor é diretamente proporcional ao tráfego. O *layout* decorrente da estratégia **intuitiva** é apresentados nas figuras 7(b)-7(d). De forma similar, o *layout* decorrente da estratégia **balanceada** é apresentado nas figuras 7(e)-7(g). Finalmente, a estratégia **gaussiana** é indicada nas figuras 7(h)-7(j).

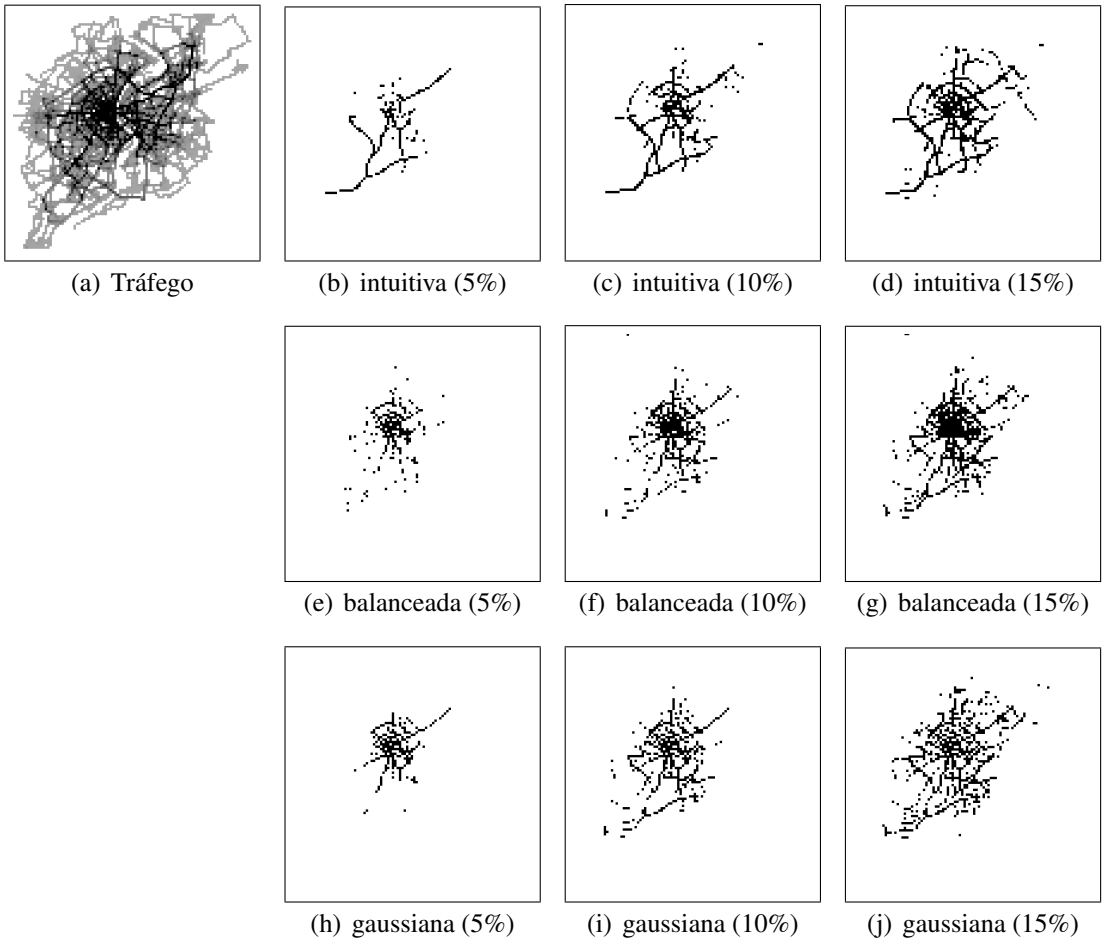


Figura 7. A figura 7(a) mostra o tráfego na cidade de Colônia, Alemanha. O *layout* decorrente da disposição de células cobertas pela estratégia intuitiva é apresentados nas figuras 7(b)-7(d). A estratégia balanceada está nas figuras 7(e)-7(g). A estratégia gaussiana é indicada nas figuras 7(h)-7(j).Note que cada coluna agrupa o mesmo cenário, facilitando a comparação entre as estratégias.

Nota-se que a estratégia **intuitiva** segue os principais corredores de tráfego da malha viária. Já a estratégia **balanceada** realiza a cobertura em círculos concêntricos a partir do epicentro de tráfego, afinal seu objetivo é maximizar a área sob a curva Delta. Já a estratégia **gaussiana** mescla características de ambas. Veja que o corredor de tráfego posterior existente na figura 7(b) não existe na figura 7(e), mas aparece na figura 7(h). Por outro lado, o *layout* geral da estratégia **gaussiana** herda a característica de espalhamento e cobertura concêntrica da estratégia **balanceada**.

7. Conclusão

Este trabalho discute o desempenho de redes veiculares em termos da métrica Rede Delta. A Rede Delta caracteriza a conectividade da rede veicular pela porcentagem de tempo de viagem em que os veículos estão conectados às unidades de beira de estrada. Em resumo, este artigo revisa o conceito da Rede Delta e estende seu estudo através da apresentação de uma estratégia baseada na distribuição gaussiana do tempo de contato entre veículos e a infraestrutura. Considera-se o rastro realístico de mobilidade veicular da cidade de Colônia, Alemanha, com a malha viária particionada em uma estrutura semelhante a uma grade de dimensões 100×100, resultando em células urbanas de dimensão $270\text{m}\times260\text{m}$ por onde trafegam mais de 75 mil veículos.

A definição da pontuação de cada célula baseia-se no número de veículos cobertos em relação à distância percorrida. Como principal benefício dessa abordagem, o padrão de cobertura resultante favorece que a maior parte da frota experimente conectividade semelhante, enquanto que veículos com baixíssima ou altíssima conectividade se tornam exceção. Os resultados demonstram que a heurística atinge seu objetivo, promovendo a distribuição de tempos percentuais de contato de veículos com a infraestrutura em acordo com a distribuição gaussiana. Como baseline, a estratégia é avaliada com a aplicação da mesma heurística usando uma distribuição de probabilidades uniforme, bem como a intuitiva deposição da infraestrutura nos locais de maior tráfego.

Como trabalho futuro, pretendemos desenvolver modelos ótimos para a estratégia aqui proposta, bem como buscar novas estratégias para atribuir pontuações para células, gerando novas estratégias de implantação.

Agradecimentos

Esse trabalho foi parcialmente financiado por CNPq, CAPES, FAPEMIG, Departamento de Tecnologia da Universidade Federal de São João del-Rei (DTECH/UFSJ) e o Departamento de Computação da Universidade Federal de Ouro Preto (DECOM/UFOP).

Referências

Bazzi, A., Masini, B. M., and Andrisano, O. (2011). On the frequent acquisition of small data through rach in umts for its applications. *IEEE Transactions on Vehicular Technology*, 60(7):2914–2926.

Cataldi, P. and Harri, J. (2011). User/operator utility-based infrastructure deployment strategies for vehicular networks. In *Vehicular Technology Conference (VTC Fall), 2011 IEEE*, pages 1–5.

Chi, J., Jo, Y., Park, H., and Park, S. (2013). Intersection-priority based optimal rsu allocation for vanet. In *Ubiquitous and Future Networks (ICUFN), 2013 Fifth International Conference on*, pages 350–355.

Katzev, R. (2003). Car sharing: A new approach to urban transportation problems. *Analyses of Social Issues and Public Policy*, 3(1):65–86.

Lee, J. and Kim, C. (2010). A roadside unit placement scheme for vehicular telematics networks. In Kim, T.-h. and Adeli, H., editors, *Advances in Computer Science and Information Technology*, volume 6059 of *Lecture Notes in Computer Science*, pages 196–202. Springer Berlin Heidelberg.

Liu, Y., Niu, J., Ma, J., and Wang, W. (2013). File downloading oriented roadside units deployment for vehicular networks. *Journal of Systems Architecture*, 59(10, Part B):938 – 946. Advanced Smart Vehicular Communication System and Applications.

Lozano-Perez, T. (2012). *Autonomous robot vehicles*. Springer Science & Business Media.

Pereira, L. A. A., Luciano, M. C., Mariz, L. L., Ramos, B. V., Pitangui, C. G., and Silva, C. M. (2018). Um modelo multiobjetivo para alocação de unidades de comunicação para distribuição de conteúdos em redes veiculares. *Workshop de Gerência e Operação de Redes e Serviços (WGRS$_S BRC$)*, 23.

Prospects, W. U. (2007). World urbanization prospects: The 2007 revision.

Sarubbi, J. F. M. and Silva, C. M. (2016). Delta-r: A novel and more economic strategy for allocating the roadside infrastructure in vehicular networks with guaranteed levels of performance. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 665–671.

Silva, C. M., Aquino, A. L. L., and Meira Jr., W. (2014). Design of roadside infrastructure for information dissemination in vehicular networks. In *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pages 1–8.

Silva, C. M., Aquino, A. L. L., and Meira Jr, W. (2015a). Smart Traffic Light for Low Traffic Conditions. *Mobile Networks and Applications*, pages 1–9.

Silva, C. M., Aquino, A. L. L. A., and Meira, Jr, W. (2015b). Deployment of roadside units based on partial mobility information. *Computer Communications*, 60(0):28 – 39.

Silva, C. M., Masini, B. M., Ferrari, G., and Thibault, I. (2017). A Survey on Infrastructure-Based Vehicular Networks. *Mobile Information Systems*, 2017, Article ID 6123868:28–56.

Silva, C. M. and Meira Jr, W. (2015). Design of roadside communication infrastructure with QoS guarantees. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, pages 439–444.

Silva, C. M. and Meira Jr., W. (2015). Evaluating the Performance of Heterogeneous Vehicular Networks. In *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, pages 1–5.

Silva, C. M., Meira Jr., W., and Sarubbi, J. F. M. (2016). Non-intrusive planning the roadside infrastructure for vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):938–947.

Silva, C. M., Silva, L. D., Santos, L. A. L., Sarubbi, J. F. M., and Pitsillides, A. (2018). Broadening understanding on managing the communication infrastructure in vehicular networks: Customizing the coverage using the delta network. *Future Internet*, 11(1).

Trullols, O., Fiore, M., Casetti, C., Chiasserini, C., and Ordinas, J. B. (2010). Planning roadside infrastructure for information dissemination in intelligent transportation systems. *Computer Communications*, 33(4):432 – 442.

Uppoor, S., Trullols-Cruces, O., Fiore, M., and Barcelo-Ordinas, J. M. (2014). Generation and analysis of a large-scale urban vehicular mobility dataset. *IEEE Transactions on Mobile Computing*, 13(5):1061–1075.

Van Audenhove, F.-J., Korniichuk, O., Dauby, L., and Pourbaix, J. (2014). *The Future of Urban Mobility 2.0: Imperatives to Shape Extended Mobility Ecosystems of Tomorrow*.

Xie, B., Xia, G., Chen, Y., and Xu, M. (2013). Roadside infrastructure placement for information dissemination in urban its based on a probabilistic model. In *Network and Parallel Computing*, volume 8147 of *Lecture Notes in Computer Science*, pages 322–331. Springer Berlin Heidelberg.

Zheng, Z., Lu, Z., Sinha, P., and Kumar, S. (2010). Maximizing the contact opportunity for vehicular internet access. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9.

Zheng, Z., Sinha, P., and Kumar, S. (2009). Alpha coverage: Bounding the interconnection gap for vehicular internet access. In *INFOCOM 2009, IEEE*, pages 2831–2835.

Análise de Vazamentos Temporais *Side-Channel* no Contexto da Internet das Coisas

Nelson G. Prates Jr.[1], **Andressa Vergütz** [1], **Ricardo T. Macedo**[2], **Michele Nogueira**[1]

[1]Centro de Ciência de Segurança Computacional (CCSC) – UFPR
[2]Depto. de Tecnologia da Informação - Campus Frederico Westphalen – UFSM

{ngpjunior,michele,avergutz}@inf.ufpr.br, rmacedo@inf.ufsm.br

Abstract. *The Internet of Things (IoT) enables common objects to be equipped with sensors, motivating new forms of applications. These objects collect data from users and the environment, making them interesting targets for attackers who want to access or manipulate them. However, even with the encrypted data, side-channel attacks exploit the devices characteristics in order to infer information that can compromise the security of the network structure. In the literature, most of the works that exploit these attacks do not consider scenarios composed of standard protocols for IoT, which exploit do not consider the identification of identical devices. This work presents a way of characterizing the identical IoT devices by exploring only the time information. For this, activities were carried out such as structuring an experimental scenario, collecting the traffic, extracting statistical characteristics and finally identifying devices. The results show that even identical devices can be identified with 100% accuracy.*

Resumo. *A Internet das Coisas (IoT) possibilita que objetos comuns sejam equipados com sensores, motivando novas formas de aplicações. Estes objetos coletam dados dos usuários e do ambiente, o que os tornam alvos interessantes para atacantes que almejam acessá-los ou manipulá-los. Entretanto, mesmo com os dados criptografados, os ataques side-channel exploram as características dos dispositivos, a fim de inferir informações que podem comprometer a segurança da estrutura da rede. Na literatura, a maior parte dos trabalhos que exploram estes ataques, não consideram cenários compostos por protocolos padronizados para a IoT, os que exploram, não consideram a identificação de dispositivos idênticos. Este trabalho apresenta uma forma de caracterizar os dispositivos IoT idênticos explorando unicamente a informação tempo. Para isso, foram realizadas atividades como estruturar um cenário experimental, coletar o tráfego, extrair características estatísticas e, por fim identificar os dispositivos. Os resultados apontam que mesmo dispositivos idênticos podem ser identificados com até 100% de precisão.*

1. Introdução

A IoT tem proporcionado a criação de ferramentas a exemplo de redes de comunicação seguras para monitorar a saúde de pessoas em suas atividades do cotidiano, informando transeuntes próximos sobre sua situação emergencial, ou ainda dispositivos capazes de predizer se há a iminência de um ataque cardíaco e assim acionar médicos ou serviços de socorro [Vergütz et al. 2017]. Com a IoT nesse processo de expansão é gerado um

significativo volume de dados sobre diversos aspectos. Como exemplo didático, a coleta de temperatura do ambiente, associada à intensidade do trânsito e a pressão arterial ou ritmo cardíaco de um motorista podem ser avaliados em tempo real e assim confirmar ou refutar correlações. Dessa forma, a IoT proporciona novos serviços, disponibiliza novos recursos, possibilita a criação de cidades mais inteligentes, além de ser atraente para os negócios devido ao volume de dados gerados que permite a melhor compreensão do comportamento de consumo, interesses pessoais, entre outros.

Todavia, devido ao volume e tipo de informação proporcionado pela IoT, ela se torna alvo dos ataques *side-channel* que se aproveitam de todo ou qualquer dado trafegado mesmo que criptografados (vazamentos *Side-Channel*) [Sayakkara et al. 2019]. Os atacantes, mediante *sniffers* que capturam o tráfego de rede, sondam os vazamentos *side-channel* como o tempo de resposta e o tamanho dos pacotes [Yan et al. 2017]. Tais vazamentos *side-channel* consistem de informações, que muitas vezes vazados de forma não intencional, podem ser classificados a fim de inferir informações que revelam características da rede e até mesmo o comportamento dos dispositivos e/ou protocolos. Além disso, a quebra da privacidade causada por esses ataques pode revelar informações e comportamento dos usuários e dos dispositivos. Por exemplo, [Srinivasan et al. 2008] inferiram informações sobre os usuários por meio de análises do tráfego em um cenário de casa inteligente (*smart-home*). Essas informações são adquiridas através da similaridade do tempo de resposta das transmissões realizadas pelos dispositivos IoT. Isso possibilita a identificação de cômodos, quantidade de residentes e até mesmo possíveis visitantes. Também, os dispositivos IoT tendem a transmitir informações cada vez mais pessoais/privadas dos usuários, como dados vitais, de posicionamento, de operações/serviços empresariais, entre outros. Entretanto, a maioria dos estudos da literatura que abordam este tipo de ataque não consideram os protocolos específicos para IoT, como 6LoWPAN e o CoAP.

Na literatura existem estudos que exploram o vazamento temporal *side-channel*. Todavia, a maioria dos trabalhos que realizam análises temporais sobre o tráfego de rede consideram os protocolos de rede tradicionais, como TCP/IP, IEEE 802.11 e HTTPs [Veysset et al. 2002, Srinivasan et al. 2008, Feghhi and Leith 2016]. Existem estudos que mencionam os dispositivos IoT [Apthorpe et al. 2017, Conti et al. 2018]. Entretanto, tais estudos também seguem os protocolos das redes tradicionais. O estudo de [Selis and Marshall 2017], apesar de considerar outro tipo de ataque, apresentou uma forma de caracterizar o comportamento de dispositivos IoT através do tempo de resposta de forma eficiente. Porém, os autores consideraram apenas o comando PING do protocolo ICMP. Finalmente, encontrou-se apenas um trabalho que considera os protocolos desenvolvidos para a IoT [Yan et al. 2017], onde os autores exploraram o comportamento de diferentes dispositivos IoT através das informações sobre o tamanho dos pacotes e tempo de resposta. No entanto, nenhum dos trabalhos encontrados na literatura diferenciam dispositivos IoT que apresentam características idênticas, *i.e.*, mesma marca, modelo, protocolos e programas em execução, através de análises temporais do tráfego de rede.

Neste sentido, este trabalho apresenta uma análise sobre o vazamento temporal *side-channel*, a fim de caracterizar e identificar dispositivos idênticos em uma rede IoT. Através desta análise pretende-se motivar o desenvolvimento de sistemas de gerência de monitoramento e segurança que necessitem da identificação de dispositivos. Além disso, motivado pela violação de privacidade e consequências causadas por tais ataques,

pretende-se destacar a relevância das informações *side-channel* na caracterização do tráfego de dispositivos mesmo idênticos. Para este fim, a análise proposta compreende a coleta do tráfego empírico de dispositivos IoT, a extração das informações *side-channel* para criação dos cenários e a identificação dos dispositivos. As informações *side-channel* consideradas englobam o `timestamp` e o tempo de resposta, bem como suas medidas estatísticas. Por meio desse conjunto de informações, esta análise apresenta um comportamento único e específico de cada dispositivo considerado.

A avaliação da caracterização e identificação propostas empregam tráfego de dados empíricos gerados e capturados através de um cenário experimental executando os principais protocolos IoT, como 6LoWPAN e CoAP. Para isso, gera-se tráfego de três dispositivos IoT Memsic Iris idênticos embarcados com sensores de iluminação e de temperatura ambiente. Na ferramenta de monitoramento de redes Wireshark[1] realiza-se a coleta do tráfego e o monitoramento da conexão do cliente CoAP. Na ferramenta de análises estatísticas RStudio[2] selecionam-se e extraem-se as características sobre as capturas de rede, como `timestamp` e tempo de resposta. Embasado nessas características, analisa-se o comportamento único de cada dispositivo IoT. Na ferramenta de mineração de dados WEKA[3], aplicam-se classificadores de aprendizagem de máquina para identificar os dispositivos. Além disso, analisa-se a eficiência de cada classificador através de métricas como acurácia e *F-Score*. Os resultados observados nas análises apontam que é possível caracterizar e classificar dispositivos idênticos, alcançando taxas de precisão de até 100%.

O restante do artigo está organizado como segue. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 detalha a análise do vazamento *side-channel*. A Seção 4 descreve a metodologia de avaliação. A Seção 5 apresenta e discute os resultados obtidos. Por fim, a Seção 6 conclui o trabalho e apresenta as direções futuras.

2. Trabalhos Relacionados

Na literatura existe uma vasta quantidade de trabalhos que analisam vazamentos *side-channel* em diferentes contextos, porém poucos consideram os protocolos padronizados para a IoT. Entretanto, os dispositivos IoT possuem características específicas como mobilidade e escassez de recursos energéticos. Devido a isso, implementar os protocolos já padronizados pela Internet, que foram projetados para dispositivos mais robustos e conexões confiáveis, se torna inviável na maioria dos cenários IoT. Em vista disso, a IETF (*Internet Engineering Task Force*), através da RFC 4944 [Montenegro et al. 2007a], RFC 4919 [Montenegro et al. 2007b], RFC 7252 [Shelby et al. 2014] e RFC 6347 [Rescorla and Modadugu 2012], padronizou os protocolos IEEE 802.15.4 (Física), 6LoWPAN (Rede), CoAP (Aplicação), DTLS (Segurança fim-a-fim), respectivamente, com o objetivo de atender aos dispositivos com recursos limitados da IoT [Prates et al. 2018]. Neste contexto, [Chen et al. 2010] e [Yan et al. 2017] apresentaram as características que podem ser extraídas através do vazamento e os potenciais ataques *side-channel* direcionados para protocolos de redes tradicionais e para o protocolo *6LoWPAN*, respectivamente. Contudo, [Yan et al. 2017] provam a existência dos Vazamentos *Side-Channel* e quais características eles podem revelar sobre a estrutura. Além disso, destacaram a carência

[1]Wireshark, https://www.wireshark.org/. Último acesso em Mar/2019.
[2]RStudio, https://www.rstudio.com/. Último acesso em Mar/2019.
[3]WEKA, https://www.cs.waikato.ac.nz/ml/weka/. Último acesso em Mar/2019.

de trabalhos que exploram este tipo de ataque considerando os protocolos para IoT. A justificativa para tal carência consiste da recente padronização destes protocolos.

Por outro lado, existem estudos que analisam o vazamento *side-channel* para inferir informações sobre os tipos de dispositivos, aplicações, sistemas operacionais, etc. [Sivanathan et al. 2018] coletaram os dados de diferentes dispositivos IoT e analisaram as características que auxiliam na descrição do comportamento dos mesmos, como números de portas e padrões de atividades. Além disso, os autores utilizaram algoritmos de classificação para identificar os dispositivos. Entretanto, os autores consideraram informações de entrada de alto nível, como a incidência de palavras chaves e o tráfego de pesquisas em servidores DNS. Em outro contexto, [Saltaformaggio et al. 2016] analisaram informações do cabeçalho IP para classificar sistemas operacionais móveis. Nas análises os autores caracterizaram o tráfego de dados por meio de métodos estatísticos e *clusterizadores*. No entanto, estes estudos consideraram somente o tráfego *web* do protocolo HTTP. [Taylor et al. 2018] exploraram as informações sobre o tamanho e a direção dos pacotes por meio de vazamento *side-channel* a fim de identificar aplicativos móveis. Para isso, os autores mitigaram o efeito de tráfego ambíguo (tráfego comum entre os aplicativos) como anúncios. Apesar dos esforços dos estudos na análise de informações sobre o tráfego de dados, eles não consideraram os protocolos específicos para IoT, focando somente nos protocolos desenvolvidos para as redes tradicionais da arquitetura TCP/IP.

Além do mais, os estudos supracitados analisam o comportamento de dispositivos, sistemas e/ou aplicações por meio de diferentes informações e características do tráfego, como tamanho dos pacotes e expressões regulares. [Yan et al. 2017] observaram as características de tamanho dos pacotes e o tempo de resposta em um cenário composto por diferentes dispositivos, onde destacaram a importância da característica tempo. A informação sobre o tempo de resposta apresenta menor quantidade de ruído quando comparada ao tamanho do pacote, e com isso, alcança uma precisão maior na classificação do tráfego. Esta importância é reforçada através do trabalho de [Srinivasan et al. 2008], onde os autores, através de análises estatísticas, utilizaram o tempo coletado do tráfego de sistemas RFID para inferir informações pessoais dos usuários. Contudo, existem estudos que apontam a possibilidade de avaliar tal comportamento considerando apenas as informações relacionadas ao tempo do pacote. Por exemplo, [Veysset et al. 2002] caracterizaram sistemas operacionais através de análises sob o tempo RTT dos pacotes do tráfego TCP. [Feghhi and Leith 2016] classificaram páginas *web* utilizando somente a informação de tempo capturada através do tráfego de dados. [Malik et al. 2017] identificaram diferentes características dos sistemas operacionais móveis para classificar as aplicações ativas. Os testes seguiram uma série de requisições do tipo PING, onde os intervalos de tempo entre as respostas foram analisados para identificar as aplicações.

Também existem estudos que exploram as informações sobre o tempo de resposta para auxiliar na defesa contra ataques de redes. [Selis and Marshall 2017] propuseram um modelo de detecção que identifica máquinas virtuais intrusas falsificando o tempo de resposta. Entretanto, apesar de ser no contexto IoT, tal estudo não emprega os protocolos IoT. Desse modo, de acordo com o melhor do nosso conhecimento, não existem estudos que analisam o comportamento de dispositivos IoT, considerando as características e protocolos específicos da IoT. Além disso, não existem estudos que diferenciem o comportamento de dispositivos IoT idênticos (mesma marca, modelo e protocolos IoT). Neste

sentido, o diferencial deste trabalho consiste em caracterizar dispositivos IoT idênticos apenas utilizando informações sobre o tempo de resposta, ou seja, apenas considerando características do vazamento temporal *side-channel*.

3. Análise de Vazamentos Temporais *Side-Channel*

Esta seção descreve a análise de vazamento temporal *side-channel* no contexto de redes IoT, a fim de auxiliar na melhoria das técnicas de gerência de monitoramento e segurança. Em particular, explora-se a informação relacionada ao tempo, adquirida através da captura do tráfego de redes IoT. Além disso, caracteriza-se e identifica-se dispositivos da mesma marca e modelo, com os mesmos protocolos e programas em execução. Para isso, as próximas subseções discutem os efeitos que as informações inferidas através do tempo podem causar no contexto da privacidade dos dados dos usuários, apresentam o cenário experimental de rede e por fim, detalham a caracterização dos dispositivos.

3.1. Discussão sobre a Informação Tempo

A informação tempo pode revelar o comportamento dos dispositivos em rede, levando a violação da privacidade dos dados. Devido ao comportamento do dispositivo se tratar de uma informação extremamente valiosa, os atacantes buscam encontrar padrões para inferir informações sobre cada dispositivo presente na rede. Estes padrões, na maioria das vezes, são explorados através do tempo de execução de toda a pilha de programas dos dispositivos. Com base na literatura, as análises sobre o vazamento temporal *side-channel* de fato revelam informações tanto das camadas inferiores no nível de *hardware* (por exemplo, sobre os sensores embarcados), como sobre as camadas superiores no nível de *software* (por exemplo, sobre os aplicativos em execução) [Yan et al. 2017].

Programas de computador contêm ramificações e laços de repetições condicionais para manipular entradas e produzir a saída pretendida. Dependendo dos valores de entrada, o caminho de execução de um programa pode diferir, resultando em um tempo de execução de programa diferente [Sayakkara et al. 2019]. Em um cenário composto por dispositivos IoT estes padrões podem se destacar, pois a escassez de recursos exige que os programas desenvolvidos sejam compostos por blocos de código menores e que envolvam operações mais específicas. Além disso, o tempo de execução desses códigos podem ser diferentes de um dispositivo para o outro devido à divergência de capacidade computacional, como poder de processamento, memória, bateria, entre outros. Alguns parâmetros estruturais dos cenários de redes também podem ser explorados, como a portabilidade dos dispositivos que motivam a geração de novas características no decorrer do tempo. Estas características também podem servir como informação para identificar dispositivos. A variável mais explorada pela literatura para este fim é o tempo de resposta.

As especificações definidas pelo protocolo 6LoWPAN exigem que os dispositivos estejam conectados a uma estação base (*gateway*). Esta configuração pode ser explorada pelos atacantes, pois o *gateway* possibilita que conexões externas à estrutura de rede sejam realizadas através de um ponto central. Entretanto, a captura do tráfego deste dispositivo, mesmo que criptografado, pode gerar brechas através da exploração do tempo de resposta. Este tempo pode ser explorado com o objetivo de inferir informações sobre o posicionamento e o comportamento dos dispositivos. No entanto, as informações capturadas também podem ser exploradas para auxiliar a gerência de monitoramento das redes.

Um exemplo prático é a necessidade de identificar sensores de monitoramento de saúde para oferecer prioridade de transmissão na rede.

Entretanto, existe um questionamento sobre a possibilidade de diferenciar dispositivo idênticos da mesma marca, modelo e com os mesmos protocolos e aplicações em execução por meio unicamente do vazamento temporal *side-channel*. Em vista disso, este trabalho propõem uma análise sobre os vazamentos temporais *side-channel*, a fim de diferenciar e caracterizar dispositivos idênticos estruturalmente. Para este fim, foi estruturada uma rede experimental para simular um cenário IoT, realizar capturas do tráfego de rede e caracterizar dispositivos. Esta coleta compreende a comunicação direta entre dois dispositivos, incluindo os atrasos e as falhas que podem ocorrer durante a troca de mensagens. O cenário compreende três dispositivos idênticos, agindo como servidores de dados. Onde nos quais, recebem requisições de um cliente que acessa a estrutura de rede através de um roteador de borda. O tráfego é capturado e a partir dele são extraídas as informações para a criação de cenários, obtidos através de características estatísticas. Por fim, a identificação dos dispositivos é realizada através do uso de classificadores.

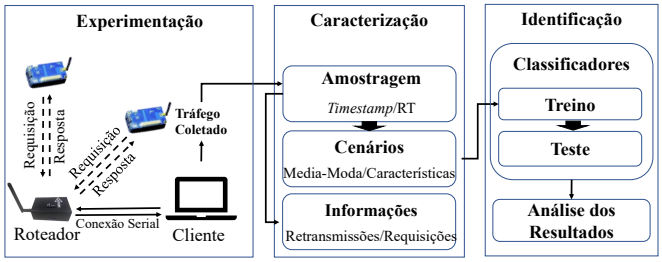


Figura 1. Etapas da Análise dos Vazamentos

Figura 2. Cenário Experimental

3.2. Cenário Experimental de Rede

A experimentação visa simular uma aplicação IoT, utilizando os principais protocolos padronizados pela IETF. Para isso, o cenário experimental é composto por quatro dispositivos IoT Memsic Iris e um computador. Os dispositivos IoT realizam diferentes operações, três deles agem como servidores e o último como estação base. Estes dispositivos IoT são equipados com *chips* Atmel's AT86RF230 compatíveis com as especificações IEEE 802.15.4 [Montenegro et al. 2007a]. Além disso, nos dispositivos podem ser acoplados placas proprietárias de sensores. As utilizadas neste cenário são as placas *MTS300CB* equipadas com sensores de iluminação e temperatura. A estação base, por sua vez, atua como um *gateway* e roteador de borda, ou seja, recebe e encaminha as requisições externas. Todos os dispositivos IoT executam o Contiki SO, com os protocolos de rede 6LoWPAN [Montenegro et al. 2007b], RPL [Thubert et al. 2017] e UDP. Para a aplicação, o protocolo CoAP [Shelby et al. 2014]. Por fim, o computador simula um cliente CoAP gerando requisições e capturando o tráfego gerado. A simulação do cliente é feita através do *framework* Californium[4] e as capturas são realizadas através da ferramenta Wireshark. O computador e o roteador de borda são interconectados através de uma interface serial/USB. A Figura 2 apresenta o cenário experimental. Considerando este cenário, também é importante afirmar que os servidores só realizam a captura dos sensores a partir do recebimento de uma requisição.

[4]Californium (Cf) https://www.eclipse.org/californium/. Último acesso em Mar/2019

3.3. Detalhamento da Caracterização e Identificação

Esta subseção detalha as fases de caracterização e identificação dos dispositivos IoT. A caracterização dos dispositivos é dividida em três etapas: *(i)* amostragem dos dados coletados, *(ii)* extração de características estatísticas para criação dos cenários e *(iii)* extração de informações extras (exemplo, taxa de retransmissões), conforme apresentado na Figura 1. A partir das coletas de tráfego da fase experimental, a amostragem desses dados segue a extração das informações sobre o `timestamp` e o tempo de resposta. Assim, cada amostra criada contém o `timestamp` de envio da requisição, o `timestamp` de recebimento da resposta e o tempo de resposta. Em seguida, extraem-se as medidas estatísticas, como média e mediana, para a criação de dois cenários de análises: um que engloba um subconjunto de medidas estatísticas e outro que envolve todas as medidas estatísticas, sendo elas média, moda, mediana, limite superior e inferior, e correlação. A entrada para a computação dessas medidas estatísticas compreende os dados extraídos na amostragem, ou seja, o `timestamp` e o tempo de resposta. Além das medidas estatísticas, a partir das amostragens de dados obtêm-se informações acerca da taxa de retransmissão e quantidade de requisições de cada dispositivo. Assim, todas essas informações em conjunto auxiliam na caracterização dos dispositivos. Em seguida, submetem-se os cenários criados com as medidas estatísticas para a fase de identificação dos dispositivos. Nesta fase, cria-se o conjunto de dados de treino e teste, para então realizar a identificação de cada dispositivo. Por fim, computam-se as métricas de desempenho para avaliação da identificação.

4. Avaliação de Desempenho

Neste estudo é analisado o vazamento de informações *side-channel* e o comportamento do tráfego considerando um cenário experimental. Particularmente, no cenário base foi coletado o tráfego de rede através da ferramenta de monitoramento de redes Wireshark. Neste cenário, um cliente realiza requisições para três dispositivos IoT Memsic Iris, que agem como servidores CoAP. A partir da captura, na ferramenta de análises estatísticas RStudio foram extraídas as características relevantes em relação ao tráfego, como tempo de resposta e tempo de envio dos dispositivos. Além disso, foram computadas as medidas estatísticas como média, mínima e moda. Tais características serviram como entrada para a caracterização do comportamento específico de cada dispositivo e para a identificação do tráfego dos mesmos. Na ferramenta de mineração de dados WEKA foram identificados os dispositivos IoT por meio de classificadores, como *Naive Bayes* e KNN. Neste sentido, as próximas subseções apresentam a metodologia de avaliação e os cenários de avaliação.

Seleção e Extração das Características

O conjunto de dados empregado nessa análise foi coletado a partir de três servidores que possuem sensores de monitoramento de iluminação e de temperatura ambiente, como detalhado na Seção 3.2. Os dados estão distribuídos em seis subconjuntos, intitulados de *Nó 1-l*, *Nó 2-l* e *Nó3-l* para os sensores de iluminação, e *Nó 1-t*, *Nó 2-t* e *Nó 3-t* para os sensores de temperatura. Cada subconjunto de dados contém uma coleta de 100.000 requisições, as quais foram divididas em 1.000 amostras de 100 requisições. A captura do tráfego foi realizada na perspectiva do **cliente**, que registrou um total de 1.226.428 pacotes transmitidos divididos em 626.428 requisições e 600.000 respostas. Além disso, como o objetivo deste estudo consiste em analisar o vazamento de informações *side-channel* de

dispositivos IoT, os dados coletados consistem de gravações normais, sem possuir nenhum tipo de ataque ou variação significante.

Na caracterização do tráfego de cada dispositivo foram considerados os dados referentes ao `timestamp` (T) e tempo de resposta (P) dos pacotes. O `timestamp` consiste do instante em que o pacote foi enviado. Enquanto o tempo de resposta se refere a diferença entre o `timestamp` de envio e o `timestamp` de resposta. Outras características dos dados de tráfego não foram consideradas, como o tamanho do pacote, pois a literatura é pobre em relação a informação tempo para os protocolos da IoT. Além do mais, a possibilidade de caracterizar o comportamento do tráfego de um dispositivo IoT apenas usando a informação tempo aponta ser um vazamento de informação *side-channel* crucial. Técnicas de gerência de monitoramento podem se beneficiar dessas informações para desenvolver novas medidas de segurança.

A fim de refinar a caracterização, foram extraídas medidas estatísticas dos dados selecionados, *i.e.,* `timestamp` de envio e tempo de resposta. Conforme a Tabela 4, as medidas estatísticas consideradas englobam a mínima (min), soma (sum), média (μ), limiar inferior (LI), limiar superior (LS), moda ($mode$), mediana (Md) e coeficiente de correlação de Pearson (r). Para algumas medidas estatísticas os valores de entrada consistem do `timestamp`, enquanto outras medidas seguem o tempo de resposta. Essas definições se embasaram no estudo de [Selis and Marshall 2017]. Para a medida do coeficiente de correlação de Pearson, foi estimado o grau de correlação entre o `timestamp` e o tempo de resposta. Com base nessas medidas, cada subconjunto possui um conjunto específico de informações que contém o tempo de resposta, `timestamp`, medidas estatísticas e rótulo. O rótulo se refere a classe alvo necessária para identificar dispositivos por meio de algoritmos supervisionados. A informação verdadeira (*ground-truth information*) sobre qual tráfego pertence a cada dispositivo foi extraída durante a coleta do tráfego no cenário experimental, o que auxiliou na rotulação da base.

Medida Estatística	Equação
Mínima (P)	$min = min(x_i)$
Soma (P e T)	$sum = \sum_{i=1}^{N} x_i$
Média (P)	$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$
Limiar Inferior (T)	$LI = \mu - 1.96\sigma$
Limiar Superior (P)	$LS = \mu + 1.96\sigma$
Moda (P e T)	$mode = freq(X)$
Mediana (P e T)	$Md = \frac{1}{2}(X_{\frac{N}{2}} + X_{\frac{N}{2}+1})$ de $sort(X)$
Coef. Correlação de Pearson (P e T)	$r = \frac{\sum_{i=1}^{N}(x_i-\mu_x)(y_i-\mu_y)}{\sqrt{\sum_{i=1}^{N}(x_i-\mu_x)^2 \sum_{i=1}^{N}(y_i-\mu_y)^2}}$

X = conjunto de dados; x_i = amostra i dos dados; N = quantidade de amostras; σ = variância; $freq$ = valor mais frequente; $sort$ = valores ordenados.

Tabela 1. Medidas Estatísticas para Caracterização do Tráfego

Cenários de Avaliação

Dois cenários de avaliação (**C1** e **C2**) foram considerados nas análises de vazamento *side-channel* dos dispositivos IoT. Nestes cenários variou-se a quantidade de medidas estatísticas do tráfego de rede utilizadas nas análises. Para isso, foi criado um subconjunto das características estatísticas apresentadas na Tabela 4, o qual engloba apenas a média e a moda do tempo de resposta. Assim, no primeiro cenário (**C1**) foi submetido o tempo

de resposta e o subconjunto de características estatísticas criado. Enquanto, no segundo cenário (**C2**) foram utilizadas todas as características estatísticas, tanto do `timestamp`, quanto do tempo de resposta. Dessa forma, é possível analisar o impacto das medidas estatísticas na caracterização do tráfego de dispositivos idênticos.

Detalhes da Identificação dos Dispositivos

A identificação dos dispositivos IoT engloba cinco algoritmos de aprendizagem de máquina amplamente utilizados e bem conhecidos na literatura [Pacheco et al. 2018]. Tais algoritmos compreendem o algoritmo clássico não paramétrico *K Nearest Neighbors* (KNN), os algoritmos baseados em árvores de decisão *Random Forrest* e *J48* (também conhecido como C4.5), a rede neural *Multilayer Perceptron*, e por fim, o algoritmo embasado na classificação bayesiana *Naive Bayes*. Esses algoritmos são aplicados em problemas de multi-classificação, ou seja, em situações que várias classes de dados precisam ser identificadas. Esta análise segue um problema de multi-classificação, pois objetiva-se identificar o tráfego de dados dos três dispositivos IoT considerados, o que justifica a escolha de tais algoritmos de classificação.

Para validar os algoritmos de classificação seguiu-se a abordagem tradicional que emprega um conjunto de dados de treino (70% dos dados) e teste (30% dos dados), a fim de treinar os modelos de classificação e computar as métricas de desempenho [Pacheco et al. 2018]. O conjunto de dados de treino possui uma captura de 100.000 requisições de dados para cada um dos seis subconjuntos, totalizando 600.000 requisições. As capturas de cada dispositivo com seu respectivo sensor foram divididas em conjuntos de 1.000 amostras, a fim de calcular as características estatísticas da Tabela 4. Assim, após pré-processamento, foi criado um conjunto de treino para ambos os sensores (temperatura e luz) com 3.000 exemplos de dados em cada, incluindo o rótulo dos três dispositivos IoT. O conjunto de dados de teste possui uma captura de 10.000 requisições de dados para cada subconjunto dos dispositivos IoT. Cada captura foi dividida em 100 amostras para computação das características estatísticas. Dessa forma, o conjunto de teste totalizou 300 exemplos de dados para cada sensor.

As métricas de desempenho consideradas envolvem a acurácia, precisão, *recall* e *F-Score* (também conhecida como *F-Measure*). Tais métricas consideram a taxa de verdadeiro positivo (VP), verdadeiro negativo (VN), falso positivo (FP) e falso negativo (FN). Dessa forma, estatisticamente a acurácia se refere a proporção de tráfego de dados classificados corretamente em relação a todas as amostras de tráfego. A precisão (p) estima a porcentagem de verdadeiros positivos dentre todos os exemplos de tráfego classificados como positivos ($VP/(VP + FP)$). O *recall* (r) ou revocação consiste da porcentagem de verdadeiros positivos dentre todos os exemplos cuja classe esperada é a positiva ($VP/(VP + FN)$). Por fim, o *F-Score* faz uma relação entre as medidas de precisão e *recall* através da estimação da média harmônica ($2rp/(r + p)$). Portanto, os resultados dos classificadores se embasam nessas quatro métricas.

5. Resultados

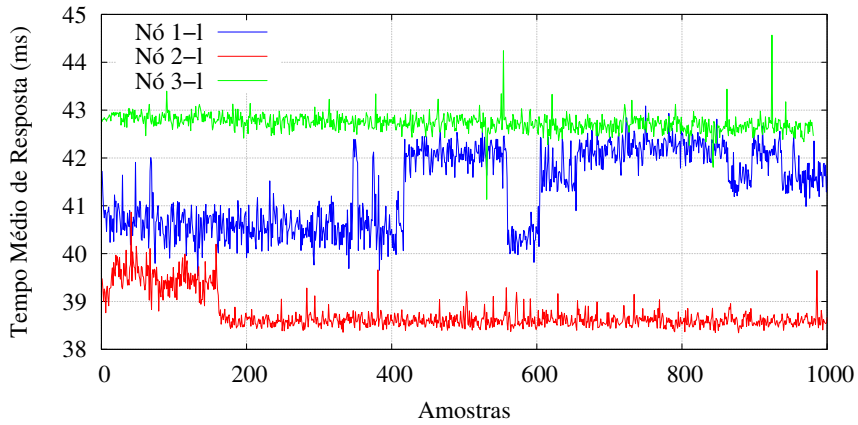
Esta seção apresenta os resultados de caracterização e identificação do tráfego de dispositivos IoT por meio do vazamento temporal *side-channel*. Os resultados seguem dois cenários de avaliação **C1** e **C2**, descritos na Seção 4. Além disso, os resultados se embasam no tráfego de três dispositivos idênticos IoT coletado em um cenário experimental.

Tais dispositivos possuem dois tipos de sensores: um sensor que monitora a iluminação e outro que monitora a temperatura ambiente. Assim, os resultados são apresentados e discutidos seguindo uma análise crítica para cada dispositivo com seu respectivo sensor.

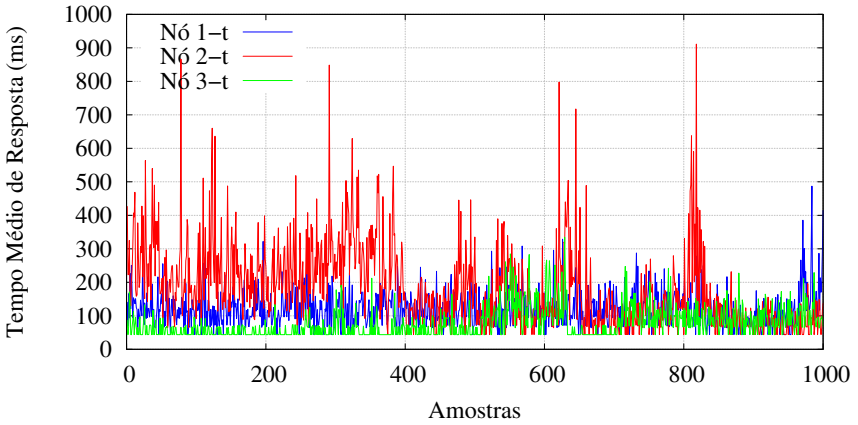
Em relação a caracterização do tráfego, a Figura 3 apresenta o comportamento do tráfego de dados dos três dispositivos IoT Memsic Irirs considerados em nossas análises. A Figura 3(a) mostra o tempo médio de resposta dos três dispositivos com sensores que monitoram a iluminação do ambiente. Enquanto, a Figura 3(b) apresenta o comportamento do tempo médio de resposta dos dispositivos com sensores de temperatura. Para cada dispositivo o tráfego de dados foi dividido em conjuntos de 1.000 amostras a fim de obter o tempo médio de resposta das requisições. Dessa forma, é possível observar um comportamento característico de cada dispositivo com seu respectivo sensor em relação ao seu tempo de resposta. Entretanto, vale observar que os dispositivos com sensores de temperatura possuem um tempo médio de resposta maior do que os de iluminação.

Mais especificamente, na Figura 3(a) o dispositivo *Nó 1* alcançou uma variação maior em seu tempo médio de resposta, atingindo valores em torno de 39 a 43ms. Em contrapartida, o dispositivo *Nó 2* apresentou uma leve queda no seu tempo médio de resposta em torno da amostra de número 200, diminuindo de 40ms para 37ms. Após essa queda, o dispositivo estabilizou com valores próximo a 37ms. Por fim, o dispositivo *Nó 3* apresentou um comportamento estável ao longo de todas as amostras de tráfego, onde seu tempo médio de resposta ficou em torno de 42 e 43ms. Desse modo, pode-se observar uma diferença clara no comportamento de cada dispositivo. Por outro lado, os dispositivos com sensores de temperatura da Figura 3(b) apresentaram um comportamento mais semelhante alcançando picos de valores de tempo médio de até 900ms para o *Nó 5*. Por se tratarem de dispositivos idênticos da mesma marca, modelo e com os mesmos protocolos e aplicações em execução, esperava-se um comportamento semelhante para ambos os sensores. Entretanto, através dos resultados observados é possível notar um comportamento específico para cada sensor. A justificativa para o sensor de temperatura alcançar valores maiores consiste do tipo de fenômeno monitorado, pois além das características dos dispositivos serem iguais, a geração e a coleta do tráfego seguiram os mesmos passos.

A Tabela 2 apresenta maiores detalhes sobre cada dispositivo com seu respectivo sensor. Como durante a geração e a coleta de tráfego de dados foi aplicado o mesmo tamanho de pacote de dados, bem como o mesmo número de requisições, a taxa de retransmissão e a média de pacotes por segundo apresentaram comportamentos similares para os três dispositivos com seus sensores. Contudo, a quantidade de requisições aumentou devido às taxas de retransmissões dos dispositivos, como por exemplo o *Nó 1-l* atingiu uma taxa de retransmissão de 7.84%. Assim, através dos resultados sobre o comportamento do tráfego de cada dispositivo observou-se que mesmo dispositivos IoT idênticos possuem um comportamento único que os diferencia no que tange o tempo médio de resposta. Tal comportamento pode ser analisado apenas considerando a característica sobre o tempo de resposta dos dispositivos. Todavia, essa informação não possui nenhuma proteção ou técnica de criptografia, o que facilita a execução de ataques *Traffic Side-Channel*. A maioria dos administradores de rede utilizam técnicas de gerenciamento de segurança que criptografam apenas o conteúdo do pacote e outras informações do cabeçalho do pacote, se abstendo de informação consideradas não tão relevantes como o tempo de reposta. Entretanto, por meio dos resultados apresentados é possível observar



(a) Dispositivos com Sensores de Iluminação



(b) Dispositivos com Sensores de Temperatura

Figura 3. Comportamento do Tempo de Resposta dos Dispositivos IoT

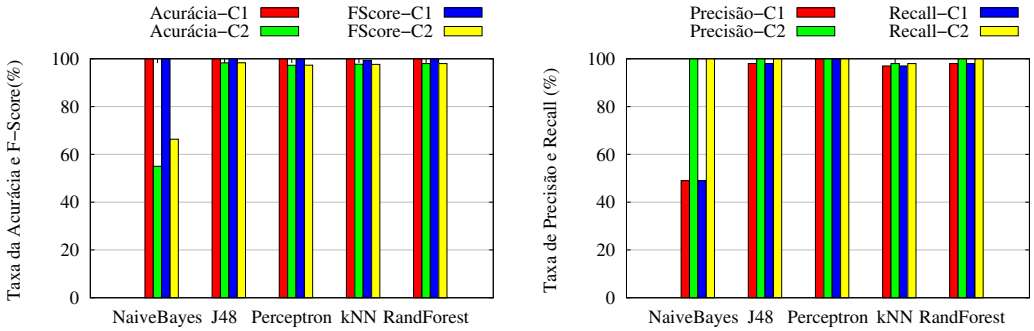
que o vazamento *side-channel* possui informações básicas que permitem caracterizar os dispositivos conectados na rede.

	Sensor de iluminação			**Sensor de temperatura**		
	Nó 1-l	*Nó 2-l*	*Nó 3-l*	*Nó 1-t*	*Nó 2-t*	*Nó 3-t*
Taxa de Retransmissão	7.84%	6.19%	3.18%	2.91%	4.98%	1.3%
Média de Pacotes/s	4.5	5.1	6.1	6.33	5.33	7.2
Tempo de Resposta Médio	41.36	38.74	46.11	119.47	182.87	77.21

Tabela 2. Detalhes dos Dispositivos IoT

Em relação a identificação do tráfego de dados de cada dispositivo IoT, os gráficos da Figura 4 apresentam os resultados referentes ao desempenho dos cinco classificadores para os sensores de iluminação. Tais resultados seguiram dois cenários de avaliação, **C1** e **C2** (detalhados na Seção 4). De acordo com o gráfico da Figura 4(a), a maioria dos classificadores alcançaram taxas de acurácia e *F-Score* próximo a 97%. Apenas o classificador *Naive Bayes* apresentou um resultado pobre para o cenário **C2**, pois neste cenário foi utilizado apenas os valores sobre a moda e a média do tempo de resposta. Porém, no cenário **C1** foram utilizadas todas as medidas estatísticas, melhorando assim o desempenho do *Naive Bayes* e dos demais classificadores. O que comprova que quanto maior o nível de detalhamento dos dados, melhor será o resultado de identificação. Um desempenho semelhante dos classificadores pode ser visto na Figura 4(b). Conforme as taxas de

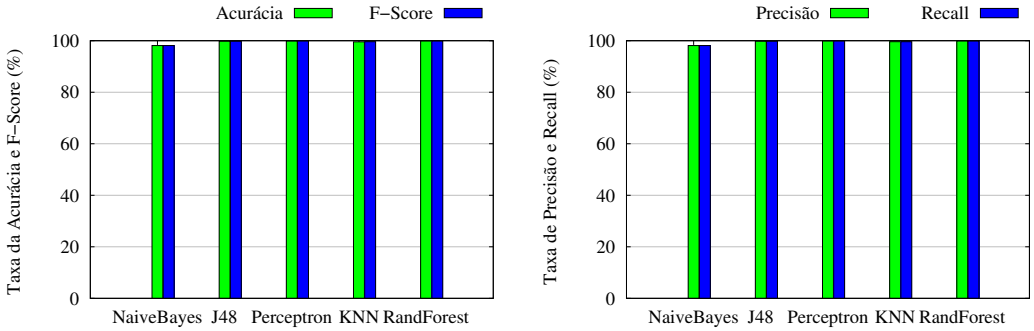
precisão e *recall*, a proporção de dispositivos classificados corretamente é extremamente alta. Dessa forma, com base no desempenho obtido por meio dos cinco classificadores comprova-se a possibilidade de identificar dispositivos IoT idênticos apenas utilizando informações relacionadas ao tempo de resposta e `timestamp` dos pacotes de dados.



(a) Acurácia e *F-Score* (b) Precisão e *Recall*

Figura 4. Desempenho dos Classificadores com Sensores de Iluminação

Para os dispositivos IoT com sensores de temperatura os resultados dos cinco classificadores também atingiram um desempenho altamente satisfatório, como pode ser observado nas Figura 5(a) e Figura 5(b). Devido aos resultados anteriores alcançarem valores ótimos para o cenário **C1**, que emprega todas as medidas estatísticas do tempo de resposta e `timestamp`, esta segunda análise considerou apenas este cenário. Nesta análise todos os classificadores alcançaram valores em torno de 90% e 99% de acurácia, *F-Score*, precisão e *recall*. Esses resultados observados reforçam as análises anteriores, o que comprova a identificação de dispositivos idênticos por meio de vazamento *side-channel*. No sentido de ataques *Traffic Side-Channel*, os atacantes podem identificar os dispositivos e assim, se passar por eles e roubar informações privadas dos usuários. Dessa forma, os resultados obtidos apontam a importância da informação tempo e a necessidade do desenvolvimento de técnicas de gerência e segurança que considerem tais informações.



(a) Acurácia e *F-Score* (b) Precisão e *Recall*

Figura 5. Desempenho dos Classificadores com Sensores de Temperatura

A Figura 6 apresenta os resultados de uma terceira análise, onde foi aplicado os dados dos três dispositivos tanto com sensores de temperatura, quanto com sensores de iluminação. Esta análise também seguiu apenas o cenário **C1**. Além disso, o tamanho do conjunto de dados de treino iniciou em 5% e aumentou gradativamente até 30%. O uso de diferentes tamanhos para o conjunto de dados de treino se embasou em um cenário que o administrador de rede precisará coletar o tráfego *online* e identificar os dispositivos o mais

rápido possível para atuar, por exemplo, contra algum ataque de rede. Assim, por meio dessa análise pode-se avaliar o desempenho dos classificadores com menor quantidade de dados de treino. Com base nos resultados, os cinco classificadores apresentaram resultados próximos a 100% de dispositivos classificados corretamente. Com isso, conclui-se a possibilidade de identificar os dispositivos mesmo com pouca quantidade de dados.

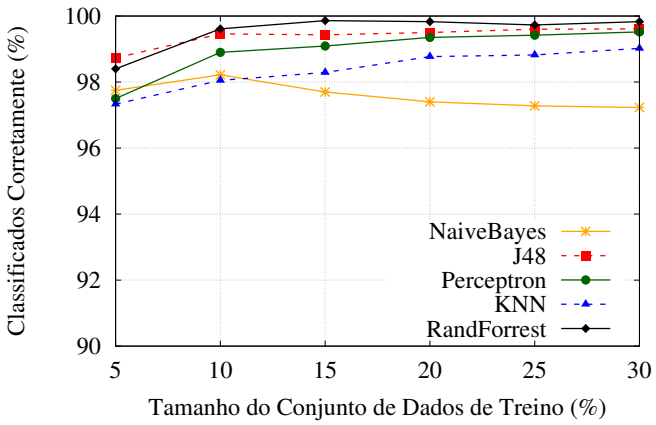


Figura 6. Taxa de Classificação considerando os Sensores de Iluminação e Temperatura

6. Conclusão

O presente artigo analisou o vazamento temporal *side-channel* no contexto de redes IoT a fim de auxiliar na melhoria das técnicas de gerência de monitoramento e segurança. Esta análise observou o comportamento de três dispositivos IoT Memsic Iris idênticos em questões de modelo, marca, protocolos e programas executados. Tal análise se embasou unicamente nas informações relacionadas ao `timestamp` e ao tempo de resposta dos dispositivos. Além disso, o tráfego de dados utilizado foi capturado por meio de um cenário de rede experimental, onde foram realizadas duas etapas de 100.000 requisições de dados por sensor. Através de profundas análises do comportamento do tráfego de dados de cada dispositivo, este trabalho confirmou a existência de características únicas e específicas mesmo de dispositivos idênticos. Além disso, este estudo mediu o desempenho de cinco classificadores na identificação do tráfego de cada dispositivo. Os resultados observados nessas análises apresentaram altas taxas de acurácia e precisão, comprovando assim uma grande eficiência em distinguir os dispositivos. Como direções futuras, pretende-se explorar outras características e com isso, desenvolver uma ferramenta de defesa que auxilie na ocultação dos vazamentos *side-channel*.

Referências

Apthorpe, N., Reisman, D., Sundaresan, S., Narayanan, A., and Feamster, N. (2017). Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic. *arXiv:1708.05044*.

Chen, S., Wang, R., Wang, X., and Zhang, K. (2010). Side-channel leaks in web applications: A reality today, a challenge tomorrow. In *IEEE Symposium on Security and Privacy*, pages 191–206. IEEE.

Conti, M., Li, Q. Q., Maragno, A., and Spolaor, R. (2018). The dark side (-channel) of mobile devices: A survey on network traffic analysis. *IEEE Commun. Surveys & Tuts.*, 20(4):2658–2713.

Feghhi, S. and Leith, D. J. (2016). A web traffic analysis attack using only timing information. *IEEE Trans. Inf. Forensics Security*, 11(8):1747–1759.

Malik, N., Chandramouli, J., Suresh, P., Fairbanks, K., Watkins, L., and Robinson, W. H. (2017). Using network traffic to verify mobile device forensic artifacts. In *2017 14th IEEE Annual Consum. Commun. & Netw. Conference (CCNC)*, pages 114–119. IEEE.

Montenegro, G., Kushalnagar, N., Hui, J., and Culler, D. (2007a). Transmission of ipv6 packets over ieee 802.15. 4 networks. Technical report, IETF.

Montenegro, G., Schumacher, C., and Kushalnagar, N. (2007b). IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. Technical Report 4919, IETF.

Pacheco, F., Exposito, E., Gineste, M., Baudoin, C., and Aguilar, J. (2018). Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. *IEEE Commun. Surveys & Tuts,*.

Prates, N., Pelloso, M., Macedo, R., and Nogueira, M. (2018). Ameaças de segurança, defesas e análise de dados em iot baseada em sdn. In *Minicursos SBSeg 2018*, chapter 1, pages 1–50. SBC.

Rescorla, E. and Modadugu, N. (2012). Datagram Transport Layer Security Version 1.2. Technical Report 6347, IETF.

Saltaformaggio, B., Choi, H., Johnson, K., Kwon, Y., Zhang, Q., Zhang, X., Xu, D., and Qian, J. (2016). Eavesdropping on fine-grained user activities within smartphone apps over encrypted network traffic. In *USENIX Workshop on Offensive Technologies)*.

Sayakkara, A., Le-Khac, N.-A., and Scanlon, M. (2019). A survey of electromagnetic side-channel attacks and discussion on their case-progressing potential for digital forensics. *Digital Investigation*.

Selis, V. and Marshall, A. (2017). A fake timing attack against behavioural tests used in embedded iot m2m communications. In *Cyber Security in Netw. Conference*, pages 1–6. IEEE.

Shelby, Z., Hartke, K., and Bormann, C. (2014). The Constrained Application Protocol (CoAP). Technical Report 7252, IETF.

Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., and Sivaraman, V. (2018). Classifying iot devices in smart environments using network traffic characteristics. *IEEE Trans. on Mobile Comput.*

Srinivasan, V., Stankovic, J., and Whitehouse, K. (2008). Protecting your daily in-home activity information from a wireless snooping attack. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 202–211. ACM.

Taylor, V. F., Spolaor, R., Conti, M., and Martinovic, I. (2018). Robust smartphone app identification via encrypted network traffic analysis. *IEEE Trans. on Informat. Forensics Security*, 13(1):63–78.

Thubert, P., Bormann, C., Toutain, L., and Cragie, R. (2017). Ipv6 over low-power wireless personal area network (6lowpan) routing header. Technical report, IETF.

Vergütz, A., da Silva, R., Nacif, J. A. M., Vieira, A. B., and Nogueira, M. (2017). Mapping critical illness early signs to priority alert transmission on wireless networks. In *IEEE Latin-American Conference on Commun.*, pages 1–6. IEEE.

Veysset, F., Courtay, O., Heen, O., Team, I., et al. (2002). New tool and technique for remote operating system fingerprinting. *Intranode Software Technologies*, 4.

Yan, Y., Oswald, E., and Tryfonas, T. (2017). Exploring potential 6LoWPAN traffic side channels. *IACR Cryptology ePrint Archive*, 2017:316.

Realização

SBC Sociedade Brasileira de Computação

LARC

Organização

.inf INSTITUTO DE INFORMÁTICA UFRGS

Apoio Institucional

RNP 30 anos

Patrocínio

Diamante

CAPES

CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico

cgi.br Comitê Gestor da Internet no Brasil

nic.br Núcleo de Informação e Coordenação do Ponto BR

BEDU TECH

Ouro

SAP

Prata

HUAWEI

DATACOM

acm sigcomm

@dyLNET TELECOM

BRDigital Telecomunicações

Bronze

FAPERGS

Google