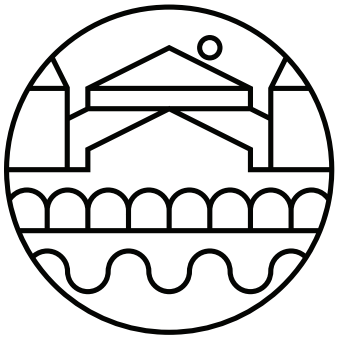
XXXV
SIMPÓSIO BRASILEIRO DE
REDES DE COMPUTADORES
E SISTEMAS DISTRIBUÍDOS

15 a 19 de maio de 2017
Belém - Pará

Anais VIII WPEIF 2017



XXXV
SIMPÓSIO BRASILEIRO DE
REDES DE COMPUTADORES
E SISTEMAS DISTRIBUÍDOS

15 a 19 de maio de 2017
Belém - Pará

Anais do VIII WPEIF 2017
Workshop de Pesquisa Experimental da Internet do Futuro

Editora
Sociedade Brasileira de Computação (SBC)

Organização
Flávio de Oliveira Silva (UFU)
José Ferreira de Rezende (UFRJ)
Ronaldo Alves Ferreira (UFMS)
Antônio Jorge Gomes Abelém (UFPA)
Eduardo Coelho Cerqueira (UFPA)

Realização
Sociedade Brasileira de Computação (SBC)
Universidade Federal do Pará (UFPA)
Laboratório Nacional de Redes de Computadores (LARC)

Capa: Catarina Nefertari (PCT-UFPA)
Produção Editorial: Denis Lima do Rosário (UFPA)

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

VIII Workshop de Pesquisa Experimental da Internet do Futuro (8: 2017: Belém, Pa).

Anais / VIII Workshop de Pesquisa Experimental da Internet do Futuro – WPEIF; organizado por Antônio Jorge Gomes Abelém, Eduardo Coelho Cerqueira, Ronaldo Alves Ferreira, Flávio de Oliveira Silva, José Ferreira de Rezende - Porto Alegre: SBC, 2017

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

1. Redes de Computadores. 2. Sistemas Distribuídos. I. Abelém, Antônio Jorge Gomes II. Cerqueira, Eduardo Coelho III. Ferreira, Ronaldo Alves IV. Silva, Flávio de Oliveira V. Rezende, José Ferreira de VI. Título.

Sociedade Brasileira da Computação

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

Laboratório Nacional de Redes de Computadores (LARC)

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

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

Diretor Executivo
Paulo André da Silva Gonçalves (UFPE)

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

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

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

Organização do SBRC 2017

Coordenadores Gerais
Antônio Jorge Gomes Abelém (UFPA)
Eduardo Coelho Cerqueira (UFPA)

Coordenadores do Comitê de Programa
Edmundo Roberto Mauro Madeira (UNICAMP)
Michele Nogueira Lima (UFPR)

Coordenador de Palestras e Tutoriais
Edmundo Souza e Silva (UFRJ)

Coordenador de Painéis e Debates
Luciano Paschoal Gaspary (UFRGS)

Coordenadores de Minicursos
Heitor Soares Ramos (UFAL)
Stênio Flávio de Lacerda Fernandes (UFPE)

Coordenadora de Workshops
Ronaldo Alves Ferreira (UFMS)

Coordenador do Salão de Ferramentas
Fabio Luciano Verdi (UFSCar)

Comitê de Organização Local
Adailton Lima (UFPA)
Alessandra Natasha (CESUPA)
Davis Oliveria (SERPRO)
Denis Rosário (UFPA)
Elisangela Aguiar (SERPRO)
João Santana (UFRA)
Josivaldo Araújo (UFPA)
Marcos Seruffo (UFPA)
Paulo Henrique Bezerra (IFPA)
Rômulo Pinheiro (UNAMA)
Ronedo Ferreira (META)
Thiêgo Nunes (IFPA)
Vagner Nascimento (UNAMA)

Comite Consultivo
Allan Edgard Silva Freitas (IFBA)
Antonio Alfredo Ferreira Loureiro (UFMG)
Christian Esteve Rothenberg (UNICAMP)
Fabíola Gonçalves Pereira Greve (UFBA)
Frank Augusto Siqueira (UFSC)
Jussara Marques de Almeida (UFMG)
Magnos Martinello (UFES)
Antonio Marinho Pilla Barcellos (UFRGS)
Moisés Renato Nunes Ribeiro (UFES)
Rossana Maria de Castro Andrade (UFC)

Mensagem dos Coordenadores Gerais

Sejam bem-vindos ao 35o Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2017) e a acolhedora cidade das mangueiras - Belém / Pará.

Organizar uma edição do SBRC pela segunda vez no Norte do Brasil é um desafio e um privilégio por poder contribuir com a comunidade de Redes de Computadores e Sistemas Distribuídos do Brasil e do exterior. O SBRC se destaca como um importante celeiro para a discussão, troca de conhecimento e apresentação de trabalhos científicos de qualidade.

A programação do SBRC 2017 está diversificada e discute temas relevantes no cenário nacional e internacional. A contribuição da comunidade científica brasileira foi de fundamental importância para manter a qualidade técnica dos trabalhos e fortalecer a ciência, tecnologia e inovação no Brasil.

Após um cuidadoso processo de avaliação, foram selecionados 77 artigos completos organizados em 26 sessões técnicas e 10 ferramentas para apresentação durante o Salão de Ferramentas. Além disso, o evento contou com 3 palestras e 3 tutoriais proferidos por pesquisadores internacionalmente renomados, 3 painéis de discussões e debates, todos sobre temas super atuais, 6 minicursos envolvendo Big Data, sistemas de transportes inteligentes, rádios definidos por software, fiscalização e neutralidade da rede, mecanismos de autenticação e autorização para nuvens computacionais e comunicação por luz visível, bem como 10 workshops.

O prêmio “Destaque da SBRC” e uma série de homenagens foram prestadas para personalidades que contribuíram e contribuem com a área. O apoio incondicional da SBC, do LARC, do Comitê Consultivo da SBRC e da Comissão Especial de Redes de Computadores e Sistemas Distribuídos da SBC foram determinantes para o sucesso do evento. A realização do evento também contou com o importante apoio do Comitê Gestor da Internet no Brasil (CGI.br), do CNPq, da CAPES, do Parque de Ciência e Tecnologia Guamá, da Connecta Networking, da Dantec Telecom, da RNP e do Google. Nosso especial agradecimento à Universidade Federal do Pará (UFPA) e ao Instituto Federal do Pará (IFPA) pelo indispensável suporte à realização do evento.

Nosso agradecimento também para os competentes e incansáveis coordenadores do comitê do programa (Michele Nogueira/UFRP – Edmundo Madeira/UNICAMP), aos coordenadores dos minicursos (Stênio Fernandes/UFPE – Heitor Ramos/UFAL), ao coordenador dos workshops (Ronaldo Ferreira/UFMS), ao coordenador de painéis e debates (Luciano Gaspary/UFRGS), ao coordenador do Salão de Ferramentas (Fabio Verdi/UFSCar) e ao coordenador de palestras e tutoriais (Edmundo Souza e Silva/UFRJ). Destacamos o excelente trabalho do comitê de organização local coordenado por Denis do Rosário.

Por fim, desejamos a todos uma produtiva semana em Belém.

Antônio Abelém e Eduardo Cerqueira

Coordenadores Gerais do SBRC 2017

Mensagem do Coordenador de Workshops

É com grande prazer que os convido a prestigiar os workshops do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) nos dias 15, 16 e 19 de maio de 2017. Tradicionalmente, os workshops abrem e fecham a semana do SBRC e são responsáveis por atrair uma parcela expressiva de participantes para o Simpósio. Como coordenador de workshops, dividi com os coordenadores gerais do SBRC a nobre tarefa de selecionar os workshops que melhor representam a comunidade e que fortaleçam novas linhas de pesquisa ou mantenham em evidência linhas de pesquisa tradicionais.

Em resposta à chamada aberta de workshops, recebemos dez propostas de alta qualidade, das quais nove foram selecionadas. Além disso, mantivemos a longa colaboração com a RNP por meio da organização do WRNP, que já é uma tradição na segunda e terça-feira da semana do SBRC. Dentre as propostas aceitas, sete são reedições de workshops tradicionais do SBRC que já são considerados parte do circuito nacional de divulgação científica nas várias subáreas de Redes de Computadores e Sistemas Distribuídos, como o WGRS (Workshop de Gerência e Operação de Redes e Serviços), o WTF (Workshop de Testes e Tolerância a Falhas), o WCGA (Workshop em Clouds, Grids e Aplicações), o WP2P+ (Workshop de Redes P2P, Dinâmicas, Sociais e Orientadas a Conteúdo), o WPEIF (Workshop de Pesquisa Experimental da Internet do Futuro), o WoSiDA (Workshop de Sistemas Distribuídos Autonômicos) e o WoCCES (Workshop de Comunicação de Sistemas Embarcados Críticos). Como novidade, teremos dois novos workshops com programação diversificada e grande apelo social, o CoUrb (Workshop de Computação Urbana) e o WTICp/D (Workshop de TIC para Desenvolvimento).

Temos certeza que 2017 será mais um ano de sucesso para os workshops do SBRC pelo importante papel de agregação que eles exercem na comunidade científica de Redes de Computadores e Sistemas Distribuídos no Brasil.

Aproveitamos para agradecer o apoio recebido de diversos membros da comunidade e, em particular, a cada coordenador de workshop, pelo brilhante trabalho. Como coordenador dos workshops, agradeço imensamente o apoio recebido da Organização Geral do SBRC 2017.

Esperamos que vocês aproveitem não somente os workshops, mas também todo o SBRC e as inúmeras atrações de Belém.

Ronaldo Alves Ferreira

Coordenador de Workshops do SBRC 2017

Mensagem dos Coordenadores do VIII WPEIF 2017

A Internet do Futuro é um conceito amplo que envolve a infraestrutura, as redes e as aplicações que provocarão um enorme impacto na sociedade. A Internet das Coisas (IoT) insere na infraestrutura um conjunto de sensores e atuadores que permitirá florescer novas aplicações e serviços que exigirão novas capacidades das redes, como baixa latência, baixo consumo de energia e garantia de entrega.

Em paralelo pesquisadores e indústria estão com olhares voltados para a próxima geração das redes móveis de telecomunicações (5G) com foco nessas capacidades.

Nesse cenário vibrante, vários grupos de pesquisa ao redor do mundo estão criando protótipos de novas arquiteturas para as próximas gerações de redes.

A Internet do futuro não será baseada em redes como as conhecemos atualmente. A infraestrutura e as redes estão se tornando mais flexíveis e reconfiguráveis, com foco nos novos requisitos que as aplicações impõem sobre as mesmas. Nesse cenário, o software assume cada vez mais um papel relevante, permitindo a programação tanto das redes quanto da infraestrutura.

Em sua oitava edição o WPEIF, possuirá duas seções técnicas cujos trabalhos envolvem o uso de plataformas experimentais, tais como FIBRE e FUTEBOL, para experimentação de novas arquiteturas de redes que utilizam do conceito de Software Defined Networking para viabilizar a experimentação e a validação.

A programação do evento será iniciada com uma palestra do Prof. Dr. Joel Rodrigues do Instituto de Telecomunicações em Portugal e do Instituto Nacional de Telecomunicações (INATEL) no Brasil.

Haverá também um painel com a participação de membros da indústria para abordar a experimentação envolvendo 5G, IoT e a Internet do Futuro.

Gostaríamos de agradecer aos autores pelo interesse no evento e pelos trabalhos submetidos, aos membros do comitê técnico de programa, do Brasil e do exterior, pela qualidade das revisões e aos coordenadores de workshops e também do SBRC pela confiança e apoio constante na organização do VIII WPEIF. Finalmente, agradecemos a toda a comunidade que participa e está sempre presente no WPEIF o que o torna um dos maiores e mais importantes workshops do SBRC.

Antônio Jorge Gomes Abelém, Flávio de Oliveira Silva, José Ferreira de Rezende

Coordenadores do VIII WPEIF 2017

Comitê de Programa

- Augusto José Venâncio Neto (UFRN)
- Cesar Augusto Cavalheiro Marcondes (UFSCAR)
- Christian Rodolfo Esteve Rothenberg (UNICAMP)
- Cristiano Bonato Both (UFRGS)
- Daniel Nunes Corujo (Universidade de Aveiro, Portugal)
- Dorgival Olavo Guedes Neto (UFMG)
- Edmundo Monteiro (Universidade de Coimbra, Portugal)
- Eduardo Coelho Cerqueira (UFPA)
- Fabio Luciano Verdi (UFSCAR)
- João Henrique de Souza Pereira (UFU)
- Joao Paulo Barraca (Universidade de Aveiro, Portugal)
- Michael Anthony Stanton (RNP)
- Natalia Castro Fernandes (UFF)
- Pedro Frosi Rosa (UFU)
- Rodrigo Sanches Miani (UFU)
- Rui Luís Andrade Aguiar (Universidade de Aveiro, Portugal)
- Sergio Takeo Kofuji (EPUSP)
- Tereza Cristina Melo de Brito Carvalho (EPUSP)
- Vinicius da Cunha Martins Borges (UFG)
- Waldir Aranha Moreira Junior (Fraunhofer, Portugal)

Sumário

VII Workshop de Pesquisa Experimental da Internet do Futuro (WPEIF)

SBRC 2017

Sessão Técnica 1

The Next Generation of the FIBRE Software Architecture

Danielle Caled[1], Tiago Salmito[1], João F. Santos[2], Leandro Ciuffo[1], José Rezende[1], Iara Machado[1]

[1]Diretoria de Pesquisa e Desenvolvimento – Rede Nacional de Ensino e Pesquisa.

[2]CONNECT Centre – Trinity College Dublin, Ireland.

{danielle.vieira, tiago.salmito, iara, leandro.ciuffo}@rnp.br[1]

jose.rezende@rnp.br[1], facocalj@tcd.ie[2]

Abstract. *The FIBRE testbed is a large-scale research facility for experimentation on Future Internet technologies. To address specific requirements of the Brazilian federation, it was decided to commence a new development phase in which FIBRE would have its architecture completely revised. This paper presents an overview of the next generation of the software architecture envisioned for the FIBRE testbed.*

1. Introduction

The FIBRE testbed[1] is a research facility focused on network experimentation operated independently by Brazilian academic institutions and lead by the National Education and Research Network (RNP). As of today (March 2017), FIBRE is a federation of 14 experimentation islands and hosts 282 users from at least 55 different organizations. Each island has a common nucleus of OpenFlow-capable switches, together with their controllers, a cluster of compute and storage servers, and optionally a cluster of wireless nodes.

As the FIBRE testbed was designed and built as an outcome of an EU-Brazil collaboration project, its original software stack is mainly inherited from the former OFELIA project[2] (2010-2013). Due to the growth of the FIBRE testbed with new experimentation islands joining the federation and also the introduction of new types of resources (e.g. wireless sensors, software-defined radios, NetFPGA servers with bare metal access, etc.), it was mandatory to redesign the FIBRE software architecture in order to deal with such a heterogeneous environment.

In early 2016, after FIBRE having financed 6 prospective short-term projects [1], and according to requirements recommended by users, it was decided to redesign the architecture of the FIBRE testbed based on 6 fundamentals[3]: full domain of the software; the adoption of a single control and management framework; improvements in user experience; authentication supported by identity federations; accounting and monitoring; and extensibility and integration of other types of resources. This decision not only led to the adoption of *cOntrol and Management Framework version 6.0* (OMF6) [3], but also to the development of a new Experimentation Portal in conjunction with a collection of services for supporting federation, accountability, and user authorization and authentication.

[1]https://www.fibre.org.br/

[2]http://www.fp7-ofelia.eu/about-ofelia/

[3]https://www.fibre.org.br/fibre-aiming-at-upgrading-to-omf6-by-2017/

This paper builds on the work described in [1] by presenting an analysis of the adoption of the OMF6 control framework. In addition, it describes the components that form the next generation of the architecture planned to be deployed in FIBRE testbed in the near future.

The rest of this paper is organized as follows: Section 2 presents the new architecture envisioned for the next phase of the FIBRE project. Section 3 and 4 describe the components developed specifically to meet the requirements of the FIBRE testbed, respectively the Experimentation Portal and Clearinghouse. Finally, in Section 5 we give some concluding remarks.

2. FIBRE's Software Architecture Redesign

The redesign of the software architecture was driven by a careful examination of the notion of experiments and testbeds in the networking field. The current architecture of FIBRE testbed employs three different control frameworks, depending on the type of resource one desires to use. The employment of multiple control frameworks hindered user experience and increased the complexity of the testbed and its operational expenses, thus the FIBRE steering committee decided to adopt OMF6 as the sole control framework running in the FIBRE testbed [1].

The basic idea behind the design of OMF6 is that "everything is a resource", this approach allowed a more natural and generic architecture. However, specific requirements of the FIBRE environment demanded the development of additional components complementary to OMF6, namely the *Experimentation Portal* and the *Clearinghouse* (CH). Figure 1 depicts the main components of the new generation of the FIBRE architecture, which we describe next.

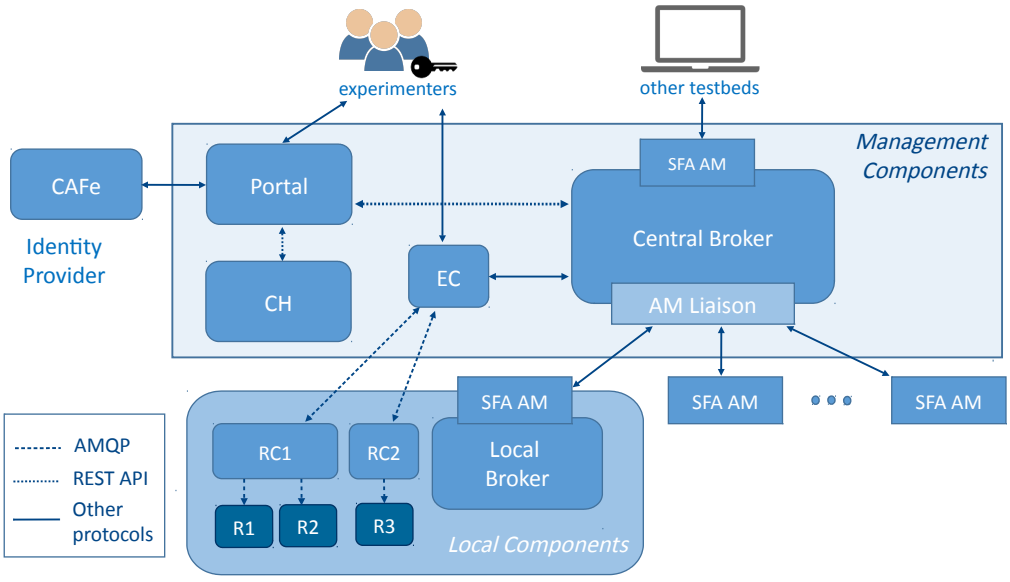


Figure 1. FIBRE's Functional Architecture

The components of the architecture are structured in two categories: *local components*, that run in each experimentation island enabling them to have direct control of local

resources, and *management components* that interact with the underlying infrastructure to expose interfaces for experimenters and other federated testbeds compatible with a *SFA Aggregate Manager API* (SFA AM).

The highest-level components of the architecture run at the management layer. Together, both Experimentation Portal and CH deal with user interaction and credential management, establishing an interface between experimenters and OMF6 components. While the Experimentation Portal was developed specifically to the FIBRE testbed, the CH was implemented according to GENI's Clearinghouse specification[4].

The *Central Broker* is responsible for aggregating and advertising resources available in the testbed. It is also responsible for resource reservation and provision in due time. The *AM Liaison* implements the core functionality of the Central Broker. It acts as the communication interface between Central Broker and Local Brokers by using their respective SFA Aggregate Manager API.

The *Experiment Controller* (EC) is the control entity responsible for orchestrating experiments described by scripts written in OMF Experiment Description Language (OEDL) [2]. A publish-and-subscribe message system is adopted for handling communications between resources and the entities interacting with them. Participants can create topics, subscribe to them and publish messages to them using AMQP[5], an open standard application layer protocol for message-oriented middlewares.

The *Local Brokers* expose a *SFA AM API* that allows discovering, aggregating and advertising local resources to the Central Broker. Each island must deploy their Local Broker to schedule and create reservations of its resources in the Experimentation Portal.

Resource Controllers (RCs) are proxies that intermediate message exchange between EC and local resources. They are responsible for controlling the life cycle of resources under their governance. They create instances of resources and send arbitrary control messages to them. These local resources may represent virtual machines, dedicated wireless enabled nodes, specialized sensors or OpenFlow resources (*R1*, *R2* and *R3* in Figure 1).

3. Experimentation Portal

The *Experimentation Portal* is a web interface built specifically to the FIBRE testbed that allows users to allocate and interact with the resources of each available island through a browser. Experimenters may create and manage shared projects, build experiments and add supported resources to them through the portal. The Experimentation Portal simplifies the use of the federation because it mediates the required interactions among CH, Broker and experimenters.

To use its services, experimenters must be authenticated. Given the diversity of users we intend to reach with FIBRE, experimenters may use the authentication granted by the Brazilian academic identity federation called *Comunidade Acadêmica Federada*[6] (CAFe), however, the portal supports local authentication to researchers who are not contemplated with a federated account. The Experimentation Portal is also designed to be

[4]http://groups.geni.net/geni/wiki/GeniClearinghouse
[5]https://www.amqp.org/
[6]http://portal.rnp.br/web/servicos/cafe

compatible with other Shibboleth identity federations.

The Experimentation Portal interacts with OMF6 Central Broker both for resource discovery and reservation through a REST API exposed by the Central Broker. OMF6 resources are only available for use during the period reserved and granted to experimenters by the local broker.

4. FIBRE Clearinghouse

The FIBRE Clearinghouse provides a collection of related services supporting the federation among experiments, resource aggregates and the underlying services of the testbed, offering services for federated authentication, user authorization and accountability. Its purpose is to manage user information and certificates, acting as the trust anchor in the federation, as it generates the credentials that grant user authorization across the other modules of the FIBRE architecture.

The CH manages projects and experimenters identities, as well as their privileges in each project context. The highest-level authentication and authorization processes are performed via the exchange of credentials between Experimentation Portal and CH.

The communication between the Experimentation Portal and the CH is again done through a REST API, which enables the management of accounts, projects and slices in the Experimentation Portal web interface. This API allows experimenters to retrieve their credentials necessary to access the OMF6 Central Broker and the OMF6 Experiment Controller.

5. Conclusion

This paper presented an overview of the next generation of the software architecture to be adopted in FIBRE testbed. This overview provided a short description of the components that form the architecture and the adoption of the OMF6 control framework. The transition to OMF6 will be as smooth as possible, with FIBRE supporting both OMF6 and the legacy control frameworks in the meantime.

The FIBRE testbed aims to be open to any researcher, professor or student interested in using the testbed for experimentation or education purposes. With a proper infrastructure to support collaboration with other projects, we expect to integrate FIBRE with other testbeds or testbed federations, more prominently the GENI and OFELIA testbed federations.

References

[1] L. Ciuffo, T. Salmito, J. Rezende, and I. Machado. Testbed fibre: Passado, presente e perspectivas. In *Anais do WPEIF 2016 Workshop de Pesquisa Experimental da Internet do Futuro*, pages 3–6, 2016.

[2] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar. Omf: A control and management framework for networking testbeds. *SIGOPS Oper. Syst. Rev.*, 43(4):54–59, Jan. 2010.

[3] D. Stavropoulos, A. Dadoukis, T. Rakotoarivelo, M. Ott, T. Korakis, and L. Tassiulas. Design, architecture and implementation of a resource discovery, reservation and provisioning framework for testbeds. In *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2015.

NovaGenesis no Ambiente FIBRE: Primeiras Impressões

Fábio Antônio Ferreira[1], **Felipe Simões Miranda**[1],
Élcio Carlos do Rosário[1], **Victor Hugo D. D'Avila**[1], **Antônio M. Alberti**[1]

[1]ICT Lab - Instituto Nacional de Telecomunicações (Inatel)
CEP 37540-000 - Santa Rita do Sapucaí, Minas Gerais, Brasil.

fantonio@gmail.com , felipe.miranda@gec.inatel.br

elcio.carlos@mtel.inatel.br, victorhdd@gmail.com e

alberti@inatel.br

Abstract. *This paper presents a preliminary analysis of the performance of the NovaGenesis architecture, a Future Internet proposal, in the FIBRE (Future Internet Brazilian Environment for Experimentation) testing environment. It describes the principles of this new architecture and a comparative analysis between its performance in a LAN network and the FIBRE environment.*

Resumo. *O presente artigo apresenta uma análise preliminar do desempenho da arquitetura NovaGenesis, uma proposta para Internet do Futuro, no ambiente de testes do FIBRE (Future Internet Brazilian Environment for Experimentation). O artigo cobre os princípios desta nova arquitetura e uma análise comparativa preliminar do seu desempenho numa rede LAN e no ambiente FIBRE.*

1. Introdução

A Internet atual foi projetada há mais de 40 anos seguindo princípios de projeto daquela época. Muitos se perguntam se o seu sucesso contínuo pode ser ameaçado, devido a falta de mecanismos de segurança, mobilidade e nomeação apropriados. Além disso, tornou-se extremamente custoso suportar as crescentes demandas por privacidade, espaços de nomeação, eficiência na distribuição de conteúdos, e outros desafios que se mostram difíceis de serem resolvidos de forma incremental. O núcleo da rede atual é TCP/IP, de difícil modificação e as propostas de IPv6 e MIPv6 têm uma lenta adoção [Czyz et al. 2014]. Para atender essas necessidades, novas funções têm sido implementadas, resolvendo o problema de forma parcial. Como resultado, paradigmas de arquiteturas totalmente novas têm sido sugeridos pela comunidade de pesquisa, com o objetivo de construir a chamada Internet do Futuro (IF ou em inglês, FI - Future Internet) [Pan et al. 2011].

A NovaGenesis (NG) foi concebida com o objetivo de criar um novo ambiente de Tecnologia de Informação e Comunicação (TIC, ou em inglês *Information and Communications Technology*) para suportar a completa, convergente e acelerada evolução tecnológica vivenciada pela humanidade. Vista como uma arquitetura de IF, a NovaGenesis é um híbrido de *name-centric*, *service-centric*, *information-centric*, *host-centric*, *software-defined*, *self-organizing* e *mobile-friendly*.

Plataformas de *testbeds*, como o FIBRE (*Future Internet Brazilian environment for Experimentation*) e o GENI (*Global Environment for Network Innovations*) [Berman et al. 2014], são fundamentais para o desenvolvimento e validação de novas arquiteturas de IF, uma vez que elas oferecem um ambiente escalável e com características próximas as reais. O principal objetivo deste trabalho é verificar o comportamento da arquitetura NG no FIBRE visando a execução de futuros testes com maior amostragem e com diferentes topologias.

Este artigo apresenta os resultados obtidos em testes preliminares da NG no ambiente FIBRE. A Seção 2 apresenta uma descrição da NovaGenesis e seu funcionamento. A Seção 3 descreve o *testebed* FIBRE. A Seção 4 detalha os testes realizados e os resultados obtidos. A Seção 5 conclui o artigo.

2. NovaGenesis

O projeto NovaGenesis [Alberti et al. 2017] foi iniciado em 2008 como um estudo de quatro anos (2008-2011) sobre o estado da arte em Internet e paradigmas emergentes de redes de comunicação. Em 2011, foi concluída a análise baseada nos conceitos de requisitos para a Internet do Futuro (IF), tecnologias e desafios. Com base no conhecimento adquirido, foi selecionado um conjunto de ingredientes promissores para resolver os desafios identificados e atender aos requisitos elencados. Além disso, trabalhou-se na busca por sinergias entre outras propostas de IF.

Os resultados deste estudo alimentaram o projeto de uma nova arquitetura chamada NovaGenesis, definindo um modelo em que o processamento e a troca de informações é tratada como um serviço. Todas as entidades são nomeadas e as ligações entre nomes são armazenadas de forma distribuída, visando atender aos requisitos de escalabilidade. Neste contexto, um nome é definido como um conjunto de símbolos (caracteres de uma linguagem) que denotam uma entidade ou um grupo de entidades, enquanto um *name binding* (NB) é um vínculo entre dois ou mais nomes. A ideia central da NovaGenesis pode ser resumida na seguinte sentença: serviços (incluindo as implementações dos protocolos) que se organizam utilizando nomes, *name bindings* (NBs) e contratos, visando atender objetivos e políticas sematicamente ricas.

A versão corrente do protótipo NovaGenesis é executada em ambiente GNU/Linux e é compatível com as atuais tecnologias da camada de enlace. Portanto, pode-se executar o protótipo sobre o Ethernet ou o Wi-Fi, e ela também pode ser executada entre dois *smartphones* usando Android e Bluetooth. Esta característica permite que a NovaGensis seja executada no ambiente FIBRE, contudo, é necessário utilizar os *raw sockets* sem a implementação do IP, de forma a permitir o encaminhamento de mensagens sobre as tecnologias de enlace.

3. FIBRE

O testbed FIBRE é uma infraestrutura de pesquisa focada em experimentação em redes e é aberta para pesquisadores. Ele é uma federação de infraestruturas experimentais, nomeadas ilhas [Salmito et al. 2014]. É o primeiro ambiente de *cyberinfrastructure* em larga escala para pesquisa em IF no Brasil. Seu objetivo é federar diversas ilhas de pesquisa em universidades brasileiras e no mundo. Uma ilha típica do FIBRE possui nós *wireless*, *switches*, NetFPGa's, *switches openflow* e sistemas de monitoramento, tudo isso

conectado ao POP mais próximo da RNP e virtualmente conectado à rede do FIBRE, a FIBREnet. A Universidade Federal do Rio Grande do Sul (UFRGS) é um exemplo, ela sedia uma Ilha do FIBRE, que está conectada ao POP-RS. O Instituto Nacional de Telecomunicações (INATEL), liga-se à Rede Nacional de Pesquisa (RNP) via Ponto de Presença de São Paulo (PoP-SP). O FIBRE utiliza o *framework cOntrol and Management Framework* (OMF) para controle e gestão dos experimentos [Rakotoarivelo et al. 2010].

4. Teste de Desempenho da NovaGenesis no FIBRE

Para este experimento foi utilizado o programa de teste da NG chamado NBSimpleTestApp, cuja operação se dá pela troca de NBs, que podem ser considerados equivalentes aos registros DNS. A operacionalização deste teste também se dá pela execução do serviço *Publish/Subscribe Service* (PSS) responsável pela comunicação publica/assina entre serviços. Através do PSS, um serviço qualquer pode expôr seus NBs a outros serviços ou descobrir como os nomes publicados estão relacionados uns com os outros de forma segura. Em ambos os cenários, FIBRE e LAN, a topologia utilizada foi estrela.

Para este experimento, foram geradas quatro diferentes quantidades de NBs, que foram publicadas no PSS e subscritas de volta pelo próprio NBSimpleTestApp. Neste procedimento, primeiramente, o NBSimpleTestApp publica todos os NBs ao PSS. Após essa etapa ele assina mil quatrocentos e quarenta NBs, com intervalo de dois segundos entre cada assinatura. Desta forma, foi possível avaliar o tempo médio de subscrição dos NBs pelo cliente e o número de subscrições feitas. É possível observar na Tabela 1 o número de subscrições no ambiente FIBRE, subscrições em LAN e subscrições esperadas para quatro cenários: duzentos mil publicações, oitocentos mil, dois milhões e oito milhões. O resultado, ilustrado pela Figura 1, apresenta uma tendência de atraso maior nos testes do FIBRE, possivelmente por ser um ambiente virtualizado e com *hardware* compartilhado. Os dados apresentam uma uniformidade na quantidade de subscrições e no atraso médio, com exceção do teste com oito milhões de publicações. Neste caso, não conseguimos estimar o atraso médio com a mesma precisão, uma vez que não foram processadas todas as subscrições previstas a cada ciclo conforme mostra a Tabela 1. A NovaGenesis ainda não possui controle de perdas de pacotes, é o caso de mensagens que não foram processadas pelo *kernel*.

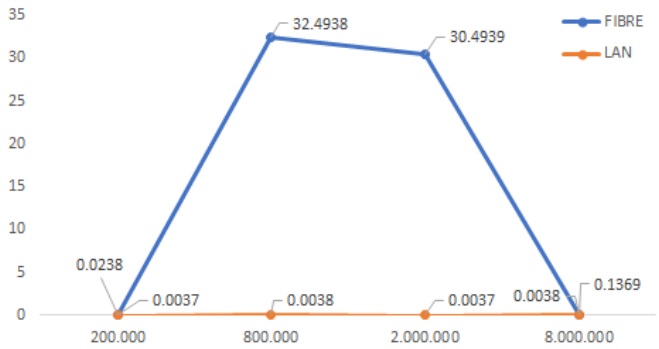


Figura 1. Resultados no FIBRE e na LAN (eixo x: número de subscrições; eixo y: tempo médio das subscrições em milissegundos).

Cenário	Subscrições FIBRE	Subscrições LAN	Subscrições esperadas
200.000	1.440	1.440	1.440
800.000	565	1.440	1.440
2.000.000	565	1.440	1.440
8.000.000	535	1.440	1.440

Tabela 1. Número de NB subscritos em cada ambiente.

5. Conclusão

Neste artigo foram realizados experimentos preliminares da NG no ambiente FIBRE. Os testes realizados mostram que o desempenho da NG no FIBRE ficou abaixo quando comparado à LAN, ou seja, não utilizando virtualização. Naturalmente, a execução em ambiente virtualizado terá desempenho inferior devido ao processo de virtualização, outro aspecto que deve ser levado em consideração é o fato do FIBRE ser um ambiente compartilhado e suscetível a atrasos por sobrecarga de processamento.

O ambiente do FIBRE possui vantagens com relação a flexibilidade e escalabilidade dos experimentos, pois recursos virtuais podem ser criados facilmente conforme a necessidade. Por este motivo, trabalhos futuros serão realizados utilizando outros cenários de testes com a NG, incluindo novas topologias e aplicações de teste e desempenho.

Agradecimentos

Este trabalho foi parcialmente financiado pela Finep, com recursos do FUNTTEL, contrato Nº 01.14.0231.00, sob o projeto Centro de Referência em Radiocomunicações (CRR) do Instituto Nacional de Telecomunicações – Inatel, Brasil. Os autores agradecem à CAPES, FAPEMIG, CNPq, RNP, IFMG Ouro Branco, SENAC Guaxupé e FINATEL.

Referências

Alberti, A. M., Casaroli, M. A. F., Singh, D., and da Rosa Righi, R. (2017). Naming and name resolution in the future internet: Introducing the novagenesis approach. *Future Generation Computer Systems*, 67:163–179.

Berman, M., Chase, J. S., Landweber, L., Nakao, A., Ott, M., Raychaudhuri, D., Ricci, R., and Seskar, I. (2014). Geni: A federated testbed for innovative network experiments. *Computer Networks*, 61:5–23.

Czyz, J., Allman, M., Zhang, J., Iekel-Johnson, S., Osterweil, E., and Bailey, M. (2014). Measuring ipv6 adoption. *SIGCOMM Comput. Commun. Rev.*, 44(4):87–98.

Pan, J., Paul, S., and Jain, R. (2011). A survey of the research on future internet architectures. *IEEE Communications Magazine*, 49(7).

Rakotoarivelo, T., Ott, M., Jourjon, G., and Seskar, I. (2010). Omf: a control and management framework for networking testbeds. *ACM SIGOPS Operating Systems Review*, 43(4):54–59.

Salmito, T., Ciuffo, L., Machado, I., Salvador, M., Stanton, M., Rodriguez, N., Abelem, A., Bergesio, L., Sallent, S., and Baron, L. (2014). Fibre-an international testbed for future internet experimentation. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC 2014*, pages p–969.

ETArch Pilot: Scaling up the Deployment of a Clean Slate Network Architecture at a Telecom Operator

Luiz Cláudio Theodoro[1,2], Pedro Damaso[1], Rogerio F. Ribeiro[1], Flávio Silva[1], Pedro Frosi[1], Alex Vaz Mendes[2], João Henrique de S. Pereira[2]

[1]Faculty of Computing – Federal University of Uberlândia (UFU)
38.408-100 – Uberlândia – MG – Brazil

[2]Innovation, Research and Development – Algar Telecom
Uberlândia, MG, Brazil

{luiz.theodoro, pedro.damaso, rogeriofr, flavio, pfrosi}@ufu.br,

{alexvaz,joaohs}@algartelecom.com.br

Abstract. *Future Internet architectures is a response from the research community to the challenges that Internet architecture faces today, such as mobility. One major issue in this area is the deployment and test of them over large scale production networks. This work scales up the deployment of the clean-slate Entity Title Architecture (ETArch) on a production network of a telecom operator. Using Virtual Tunnel (VTun) as an overlay is was possible to connect different users in different cities at Minas Gerais state in Brazil. ETArch Pilot shows the feasibility to move toward future Internet deployment in order to bring new services and applications to users.*

1. Introduction

As the Internet has become fundamental to a huge volume of worldwide activities, there is a need to refine the architecture proposed since the beginning of its operation. Trying to find solutions, researchers around the world have been struggling to propose new architectural models, new protocols using a clean-slate approach or evolution the current ones considering the same network architecture [Bronzino et al. 2013].

To evaluate new proposals, validation in an environment close to the one in real world is crucial to verify performance, restrictions and benefits when compared to the current network architecture. However, this evaluation is really complicated to be conducted on real production networks considering security aspects and also possible out of service situations.

This work extends a previous one [Claudio et al. 2015] and its goals is to scale up the deployment of a Software-Defined Networking (SDN) based clean-slate network architecture, named ETArch, in a real network managed by a telecom operator, namely Algar Telecom.

To support a growing number of users and bypass the different access technologies, a tunneling approach, named Virtual Tunnels (VTun), was used. By using VTun, it was possible to connect several Algar Telecom customers located in different cities.

This work is organized as follows: Section 2 describes the scale up of ETArch

deployment at the current network of the telecom operator. Section 3 describes the experiments conducted and finally, Section 4 presents some concluding remarks.

2. ETArch Pilot Scale Up

Proposed by our research group, ETArch has a natural match with SDN, since both share the concept that the control plane is separated from the data plane. ETArch supports several requirements from current applications such as quality of experience (QoE) during mobility [Silva et al. 2014] and multicast [Amaral Gonçalves et al. 2014].

This work proposes the scale up of the deployment of ETArch by using a real telecom network and their customers, which are geographically distributed in the operator's network.

To accomplish this, it was necessary to establish a Layer 2 connection between customers and to have an fully OpenFlow capable infrastructure. Since this last condition was not satisfied in the operator infrastructure, then a tunneling technique was used. The VTun software was chosen [VTUN 2016].

The Virtual Tunnels (VTun) is a software that act in the client/server mode, and is capable to accomplish a point to point connection between the involved hosts.

For the traffic to be tunneled by the VTun, one host needs to act as a server, opening a socket in the system and listening to the port 5000. When a client connects to this service, one virtual interface is created in the operating system. That virtual interface is the *access bridge* to the created tunnel.

Compared to the previously published work [Claudio et al. 2015], the relevant improvement occurred in the simplification of the tunneling process that occurred due to the use of VTun instead of GRE. That evolution brought to us a major simplicity since the VTun offers less complexity to accomplish the scale up of ETArch deployment. When using GRE, it is necessary to configure the client's modem in *bridge mode*.

By using VTun to produce the tunnel, it was possible to do a smooth deployment and use of ETArch based applications. With this approach, there was no need to change the customer's modem operation mode and in this case the only requirement to use ETArch based applications was to be a customer of the telecom operator. It happens because when the VTun is started, one virtual interface is created in the operating system, which is then responsible to carry the traffic through that interface and also for the created tunnel.

3. ETArch Pilot Experimental Evaluation

To conduct the tests, VTun was used to create the virtual tunnels between the hosts, allowing ETArch based applications to send data to all the hosts connected to the same workspace.

In the evaluation scenario, a concentration tunnel host was created. The function of this machine was to host the VTun in server mode and to receive all the client's connections from the ETArch architecture. In this scenario, only the VTun concentrator machine was acting like an OpenFlow switch. This made the configuration and troubleshooting process easier since there is only one OpenFlow switch in the topology. The topology for the tests conducted under this scenario is described in detail by the Figure 1.

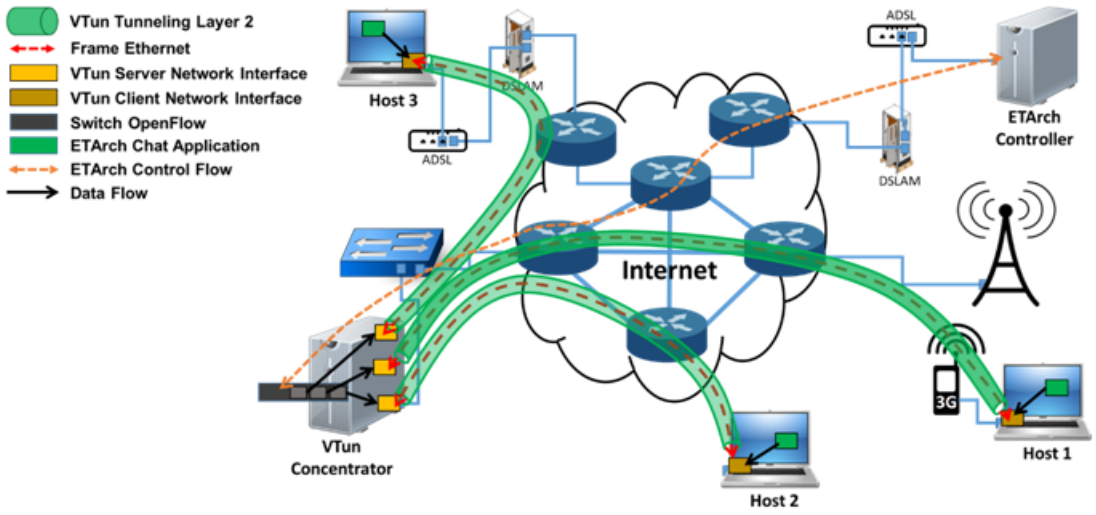


Figure 1. Communication between multiple entities using VTun.

In order to verify the overhead based on packet capture in the test environment, it was possible to identify that the VTUN encapsulation process caused an overhead of about 50.4%, equivalent to 56 bytes in each packet generated by the ETArch application. The overhead of the VTun tunneling versus the packet size ranges between 82.35% to 3.73% considering a total of 1500 Bytes to each packet.

Using the scenario, presented in Figure 1 it was possible to extend the number of entities in an unlimited geographical area. For this reason, we made a test with more than 40 users connected, distributed in a radius of 600 kilometers from the city of Uberlândia, according to Figure 2(a) which shows the localization of the machines performed by the chat clients. Most of the users are located in the city of Uberlândia as can be seen partially in Figure 2(b).

4. Concluding Remarks

This work scaled up the deployment of a clean-slate SDN based network architecture, named ETArch, in the real infrastructure of a network operator, named ALGAR Telecom, with little intervention in the customer environment.

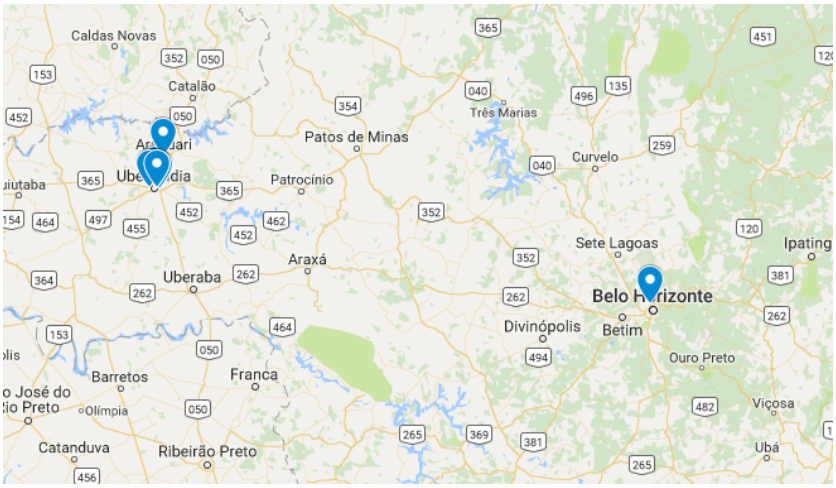
By using VTun to support the tunneling process and to tackle the interconnection issues with the infrastructure, it was possible to use an ETArch based application by several customers located in different cities inside the operator coverage area. The chat application uses natives ETArch's capabilities to support multicast and mobility.

The work demonstrates the feasibility to deploy new network architectures in parallel with current ones and go towards future Internet deployment.

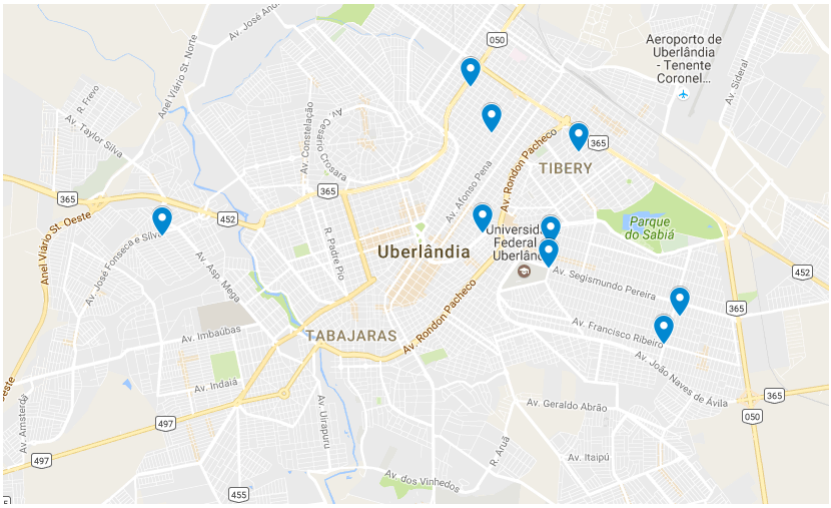
References

Amaral Gonçalves et al., M. (2014). Multicast traffic aggregation through entity title model. pages 175–180. [retrieved: Mar, 2017].

Bronzino, F., Nagaraja, K., Seskar, I., and Raychaudhuri, D. (2013). Network service abstractions for a mobility-centric future internet architecture. In *Proceedings of the*



(a) Different cities in a distance of 600 km between peers



(b) Experimentation scenario with different entities in Uberlândia

Figure 2. Geographical distribution of the Entities.

eighth ACM international workshop on Mobility in the evolving internet architecture, pages 5–10. ACM.

Claudio et al., L. (2015). Entity Title Architecture Pilot: Deploying a Clean Slate SDN Based Network at a Telecom Operator. pages 144–149. [retrieved: Mar, 2017].

Silva, F., Castillo-Lema, J., Neto, A., Silva, F., Rosa, P., Corujo, D., Guimarães, C., and Aguiar, R. (2014). Entity title architecture extensions towards advanced quality-oriented mobility control capabilities. In *2014 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6.

VTUN (2016). VTun - Virtual Tunnels over TCP/IP networks. [retrieved: Mar, 2017].

Busca de caminhos como serviço em vSDNs

André Bahia[1], **Pedro Mourão**[1], **Billy Pinheiro**[1], **Antônio Abelém** [1]

[1]Grupo de Estudos em Redes de Computadores e Sistemas Distribuídos - GERCOM
UFPa – Belém –PA
Caixa Postal 470 – 66075-110 – Belém – PA – Brasil

{andrebahia, billy, abelem}@ufpa.br, pedro.araujo@itec.ufpa.br

Abstract. *Virtual Software Defined Networks (vSDNs) were the combination of SDN and Virtualization. In this context, the hypervisor is responsible for managing the physical network, while the traditional SDN network maintains this function in the controller. This article introduces Search Path, a graph-based path finder that avoids unnecessary translations between the hipervisor de rede and the controllers in the vSDN context. The slice forwarded by the hipervisor de rede is received by Search Path and redesigned in the form of graphs, Facilitating the development of specific routing logics for each network, through the manipulation of graphs.The tests performed demonstrate that Search Path has better results than traditional hypervisors and controller.*

Resumo. *As Redes virtuais Definidas por Software (vSDNs) são a junção de SDN e Virtualização. Neste contexto, o hipervisor de rede é responsável pela gestão da rede física, enquanto a rede SDN não virtualizada mantém esta função no controlador. Este artigo apresenta o Search Path, um buscador de caminhos baseado em grafos que evita traduções desnecessárias entre o hipervisor de rede e os controladores no contexto de vSDNs. A fatia encaminhada pelo hipervisor de rede é recebida pelo Search Path em forma de grafos, facilitando o desenvolvimento de lógicas de encaminhamento específicas para cada rede, através da manipulação de grafos. Os testes realizados demonstram que o Search Path possui resultados melhores quando comparado com hipervisor de rede e controlador tradicionais.*

1. Introdução

A Virtualização de Rede (*Network Virtualization* - NV) habilita o compartilhamento do substrato físico entre diferentes instâncias virtuais, sendo este substrato formado por redes de computadores [Blenk et al. 2016] que podem estar distribuídas geograficamente.

No contexto de Redes Definidas por Software (*Software Defined Networking* - SDN), o hipervisor de rede organiza a arquitetura da rede em fatias e entrega cada fatia a um controlador, assim, permite que diferentes controladores usem a mesma rede física enquanto mantém o isolamento lógico entre eles.

O Search Path (SP) tem como objetivo simplificar as funções exercidas por um controlador em vSDNs, retirando do controlador as tarefas que são executadas pelo hipervisor de rede e criando um buscador de caminhos.

O restante deste artigo está organizado da seguinte forma: A Seção 2 apresenta as funções dos controladores e virtualizadores. A Seção 3 descreve o *Search Path*. Por fim, a conclusão e os trabalhos futuros são apresentados na Seção 4.

2. Trabalhos Relacionados

Esta seção apresenta os trabalhos relacionados as funções exercidas por controladores e virtualizados em vSDNs. As funções exercidas pelos controladores SDN [Stancu et al. 2015] são:

- Verificar o estado da rede: O controlador verifica o comportamento da rede, identificando as cargas dos enlaces e criando a topologia da rede;
- Definir o encaminhamento de dados: O controlador é responsável pela criação e manutenção das regras *OpenFlow* que são encaminhadas para os *switches*.

A Tabela 1 faz a comparação entre os diferentes controladores NOX[1], Onos [Berde et al. 2014], Beacon [Erickson 2013], Opendaylight [Medved et al. 2014] e Ryu[2] com desempenho de acordo com Andrade [Andrade et al. 2016].

Tabela 1. Tabela Comparativa dos controladores

	Utiliza grafo	Desempenho	Independente de Versão Openflow
NOX	Não	Moderado	Não
Ryu	Não	Ruim	Não
Beacon	Não	Bom	Não
Floodlight	Não	Bom	Não
Opendaylight	Não	Moderado	Não
Onos	Parcialmente	Moderado	Não

As funções atualmente exercidas pelos hipervisors de rede [Blenk et al. 2016] são:

- Dividir a rede e dar a cada controlador uma fatia (*slice*). O controlador possui conhecimento apenas da fatia da rede alocada para ele, enquanto o hipervisor de rede tem a visão global da rede;
- Mapear as topologias físicas e de todas as fatias, uma tarefa maior que a do controlador, que gerencia apenas a própria fatia;

São exemplos de hipervisors de rede o *OpenVirteX* [Al-Shabibi et al. 2014], e *Graph Virtualization Layer* (GVL)[3] [Pinheiro 2016], o último utilizado neste trabalho por reduzir a troca de mensagens entre a camada que controla o *slice* e a camada de virtualização. Através da análise das funções executadas pelos controladores e hipervisors de rede é possível identificar que a presença do hipervisor de rede, um requisito das vSDNs para acabar com a duplicação nas funções executadas na rede.

3. Search Path (SP)

O *Search Path* é um buscador de caminhos para vSDNs, que usa abstração de grafos e permite uma visão mais abrangente das redes através do dissociação com a linguagem

[1]https://github.com/noxrepo/nox
[2]Disponível em https://osrg.github.io/ryu/
[3]Disponível em https://gitlab.com/gercom/gvl/

de rede utilizada (OpenFlow). Usando como base as funções dos hipervisors de rede e controladores mapeadas na Seção 2, é possivel definir os requisitos desejáveis para um buscador vSDN, são eles:

- Receber informações do hipervisor de rede em vez de coletá-las. Para cada alteração na estrutura da rede, o hipervisor deve mapear a alteração e reencaminhá-la para o buscador, evitando fluxos de dados desnecessários. Como o SP é usado em vSDNs, ele suporta esse recurso.
- Receber dados do hipervisor sobre a entrada de novos membros na rede: Não é função de um controlador para vSDNs mapear a entrada de novos elementos da rede, mas sim receber essas informações do hipervisor. O *Search Path* usa os dados do hipervisor de rede e não faz varreduras no *slice*, uma vantagem do SP sobre os controladores.
- O buscador deve ser responsável apenas pela criação do caminho onde os dados devem prosseguir, deixando de ser responsável pela criação de regras nos equipamentos, o SP usa os conceitos de grafos e o mapeamento já feito pelo hipervisor de rede para a busca dos caminhos e posterior inserção deste no hipervisor de rede.

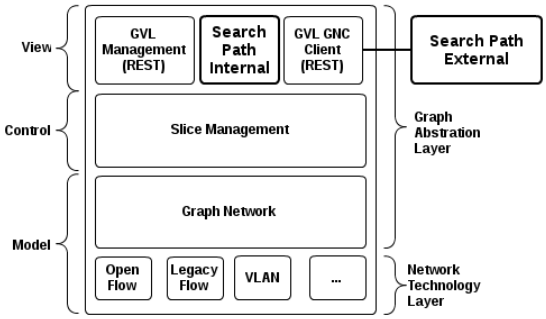


Figura 1. Definição do SPi e SPe em vSDN

A Figura 1 mostra em destaque a arquitetura do *Search Path internal*, para testes de menor caminho, e a *Search Path external*, que deverá ser utilizada em aplicações mais complexas, como *firewall*, por exemplo. As partes não destacadas compõem a arquitetura do hipervisor de rede GVL, utilizado neste trabalho para validação da proposta.

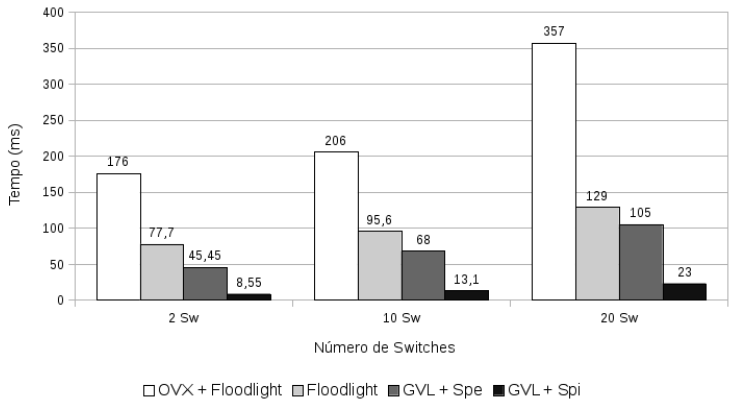


Figura 2. Gráfico entre as diferentes arquiteturas

Os testes consistiram em verificar o tempo de resposta do primeiro pedido ICMP utilizado em uma rede de topologia linear, para isso utilizamos a ferramenta Ping. Apenas o primeiro pacote atinge a camada de controle, os restantes seguem o fluxo das regras nos *switches*. A Figura 2 mostra os resultados dos testes do SP interno e SP externo utilizando o hipervisor de rede GVL, hipervisor de rede capaz de exportar a topologia dos *slices* sem usar Openflow. Nos testes tanto o *SPe* quando o *SPi* apresentaram melhores resultados que o *OpenVirtex* com o *Floodlight* e até mesmo no experimento usando apenas o *Floodlight*, ou seja, sem a camada de virtualização.

4. Conclusão e Trabalhos Futuros

Este artigo apresentou o *Search Path*, um buscador de caminhos para vSDNs que utiliza os conceitos de grafos para diminuir a troca de mensagens *OpenFlow* e apresenta ganhos de desempenho quando comparado ao *OpenVirtex* com *Floodlight*. Novas topologias devem ser utilizadas em novos testes de desempenho nos trabalhos futuros para comprovar a escalabilidade da solução.

Referências

Al-Shabibi, A., De Leenheer, M., Gerola, M., Koshibe, A., Parulkar, G., Salvadori, E., and Snow, B. (2014). Openvirtex: Make your virtual sdns programmable. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, pages 25–30, New York, NY, USA. ACM.

Andrade, L., Borba, M., Ishimori, A., Farias, F., Cerqueira, E., and Abelém, A. (2016). On the benchmarking mainstream open software-defined networking controllers. In *Proceedings of the 9th Latin America Networking Conference*, LANC '16, pages 9–12, New York, NY, USA. ACM.

Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O'Connor, B., Radoslavov, P., Snow, W., and Parulkar, G. (2014). Onos: Towards an open, distributed sdn os. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, pages 1–6, New York, NY, USA. ACM.

Blenk, A., Basta, A., Reisslein, M., and Kellerer, W. (2016). Survey on network virtualization hypervisors for software defined networking. *IEEE Communications Surveys Tutorials*, 18(1):655–685.

Erickson, D. (2013). The beacon openflow controller. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, pages 13–18, New York, NY, USA. ACM.

Medved, J., Varga, R., Tkacik, A., and Gray, K. (2014). Opendaylight: Towards a model-driven sdn controller architecture. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*.

Pinheiro, B. (2016). *Uma Abordagem SDN para Virtualização de Redes*. PhD thesis, Universidade Federal do Pará.

Stancu, A. L., Halunga, S., Vulpe, A., Suciu, G., Fratu, O., and Popovici, E. C. (2015). A comparison between several software defined networking controllers. In *2015 12th International Conference on Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS)*, pages 223–226.

VII Workshop de Pesquisa Experimental da Internet do Futuro (WPEIF)

SBRC 2017

Sessão Técnica 2

Proposta e Implementação de um *Framework* de Controle para *Testbeds* Federados que Integram Nuvem e SDN

Isabella de A. Ceravolo[1], Diego G. Cardoso[1], Cristina K. Dominicini[1]
Rodolfo da S. Villaça[1], Moisés R. N. Ribeiro[2], Magnos Martinello[1]

[1]Programa de Pós-Graduação em Informática (PPGI)
[2]Programa de Pós-Graduação em Engenharia Elétrica (PPGEE)
Universidade Federal do Espírito Santo (UFES) – Vitória/ES

isabella.ceravolo@aluno.ufes.br, dgiacomellic@gmail.com
cristina.dominicini@ifes.edu.br, rodolfo.villaca@ufes.br
moises@ele.ufes.br, magnos@inf.ufes.br

Abstract. *With the growth of Future Internet research, the demand for platforms for experimentation has grown. The FUTEBOL project aims to support research, education and innovation through the development of testbeds that allow an integrated experience between cloud computing and packet-switched, optical and wireless networks. To this end, this paper proposes a control framework architecture for testbeds that incorporates programmability and flexibility to standard technologies, such as OpenStack and OpenFlow, reducing the gap between proof of concept and production environments.*

Resumo. *Com o crescimento das pesquisas em Internet do Futuro cresceu a demanda por plataformas para experimentação. O projeto FUTEBOL objetiva apoiar pesquisa, ensino e inovação através do desenvolvimento de testbeds que possibilitam a experimentação integrada entre computação em nuvem e redes comutadas por pacote, redes ópticas e sem fio. Para esse fim, este trabalho propõe uma arquitetura de framework de controle para testbeds que incorpora programabilidade e flexibilidade à tecnologias e padrões adotados comercialmente, tais como OpenStack e OpenFlow, reduzindo a lacuna entre provas de conceito e os ambientes de produção.*

1. Introdução

Para contribuir com o avanço da pesquisa e inovação em redes e telecomunicações, pesquisadores e empresas desenvolvem novos protocolos, arquiteturas e aplicações. Uma etapa importante nesse processo é a realização de provas de conceito, o que necessita de um ambiente onde se possa reproduzir com fidelidade as condições desejadas de latência, largura de banda e escala. Os *testbeds* para experimentação de projetos de Internet do Futuro visam suprir essa necessidade por meio de plataformas que oferecem programabilidade e flexibilidade que, geralmente, não são encontradas em ambientes comerciais de computação em nuvem ou na Internet [Berman et al. 2015].

Com a crescente demanda por processamento e rede que deve ser atendida pela nuvem, pesquisadores e indústria têm se voltado para o desenvolvimento de novas soluções relacionadas à orquestração, virtualização de funções de rede e convergência entre redes móveis e ópticas. Para possibilitar a exploração dessas novas soluções, é necessário o desenvolvimento de tecnologias para apoiar a condução de experimentos em provas de conceito.

Nesse contexto surge o projeto FUTEBOL[1], que visa fomentar a pesquisa, ensino e inovação em redes e telecomunicações, habilitando a experimentação integrada em redes ópticas e sem fio. Dentre os objetivos do FUTEBOL estão a implantação de *testbeds* e o desenvolvimento de um *Control and Management Framework* (CMF) federado para controle dos *testbeds*. A federação do projeto adota uma arquitetura heterogênea, em que cada *testbed* tem liberdade para selecionar seu CMF.

Neste trabalho, apresentamos uma proposta de arquitetura para o CMF que controlará recursos que serão oferecidos pelo *testbed* FUTEBOL na Universidade Federal do Espírito Santo (UFES). A proposta visa preencher a lacuna existente entre as plataformas de mercado e as plataformas de experimentação. De um lado, grande parte dos CMFs existentes não utilizam plataformas comerciais de nuvem[2]. Do outro lado, nuvens comerciais não oferecem o nível de programabilidade da infraestrutura que é desejado por experimentadores. Assim, o CMF proposto integra a flexibilidade exigida pela federação com a robustez de um ambiente similar ao encontrado em produção.

Trabalhos similares à proposta aqui apresentada já foram desenvolvidos pelo projetos OpenGENI *rack*[3] e KOREN[4]. Entretanto, a solução OpenGENI possui alto custo e baixa flexibilidade, dificultando sua adoção no contexto apresentado. O KOREN não disponibiliza documentação suficiente para a reprodução de seu CMF.

2. Análise de Requisitos

Como membro da federação FUTEBOL, o *testbed* na UFES precisa respeitar decisões tomadas em conjunto pelos participantes do projeto. Uma das decisões é que os *testbeds* do projeto sejam integrados à federação Fed4FIRE. Assim, o CMF deve ser compatível com a arquitetura *Slice-based Federation Architecture* (SFA). Como trata-se de uma federação em que cada *testbed* é operado por uma instituição diferente e tem autonomia para controlar os recursos, o SFA é essencial para promover a consistência e interoperabilidade na descoberta, reserva e provisionamento de recursos [Sitaram et al. 2016]. Outros requisitos advindos da integração com o Fed4FIRE são: i) Suporte ao OML como ferramenta de instrumentação de experimentos; ii) Dispor de ferramenta para monitoramento do *testbed*; e iii) Suporte a ferramenta de controle de experimento.

Além disso, o CMF aqui apresentado segue as premissas de software aberto, com a intenção de proporcionar uma solução que pode ser replicada com baixo custo. Por isso, evitamos o uso de soluções proprietárias para controle dos recursos físicos. No *testbed* da UFES, serão oferecidos recursos de computação e equipamentos de redes comutadas por pacote, ópticas e sem fio. Para controlar esses recursos, utilizaremos tecnologias abertas, porém padrão de mercado (OpenFlow, por exemplo), pois acreditamos que isso favorece a consistência dos resultados obtidos pelos experimentos com as condições encontradas em ambientes de produção.

3. Projeto e Implementação do CMF do FUTEBOL

Com base nos requisitos mencionados, a Figura 1 mostra a arquitetura proposta para o CMF desenvolvido na UFES. A *Camada de Federação* atua como mediadora entre as

[1]http://www.ict-futebol.org.br/

[2]Por exemplo, o OMF (http://mytestbed.net/) e o OCF (http://fp7-ofelia.github.io/ocf/).

[3]http://www.opengeni.net/

[4]http://www.koren.kr/koren/eng/

requisições vindas da federação e o componente *Orquestrador*, que interage com os controladores dos recursos físicos. Essa camada acessa diretamente a base de dados que armazena informações sobre os recursos reservados. Essas informações, além de possibilitarem o acesso externo via SSH (*Secure SHell*) às máquinas virtuais, também são utilizadas pelo *Orquestrador*. Como o *testbed* oferecerá uma plataforma de computação em nuvem, com equipamentos de rede programáveis seguindo o conceito de *Software Defined Networks* (SDN), teremos dois *Aggregate Managers* (AMs): um para a nuvem e outro para recursos SDN. Para apoiar o desenvolvimento dos AMs, foi selecionada a implementação de referência das interfaces SFA desenvolvida pelo projeto GENI, o *GENI Control Framework* (GCF) [Kim and Lee 2012]. Dessa forma, cada AM, além de utilizar o GCF, também possui um componente, chamado de *Despachante*, responsável por traduzir as chamadas SFA em chamadas para a API *RESTful* do *Orquestrador* do *testbed*.

O *Orquestrador* do *testbed* tem a função de definir quais recursos devem ser empregados no atendimento das requisições, por meio de otimizações realizadas internamente. A *Camada de Orquestração* é fundamental, pois permite a manipulação integrada e eficiente de recursos e promove o desacoplamento entre os mecanismos que habilitam a federação e as tecnologias que gerenciam os recursos.

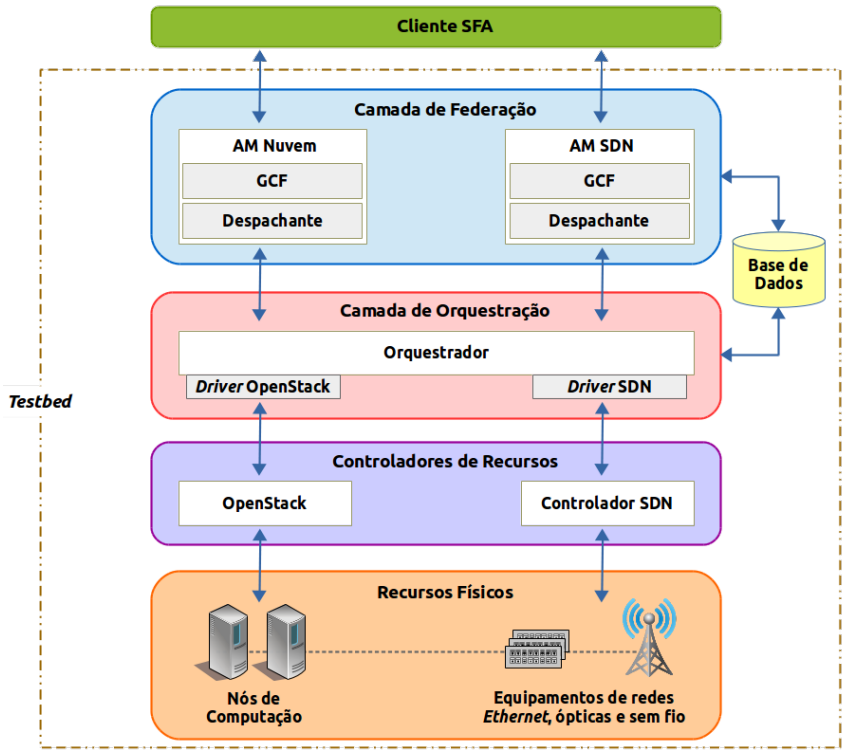


Figura 1. Arquitetura do *framework* de controle do *testbed*.

A camada de *Controladores de Recursos* contém as plataformas utilizadas na gestão dos recursos físicos. Para gerir os recursos de computação em nuvem, selecionamos o OpenStack, definido como um sistema operacional de nuvem [Sitaram et al. 2016]. Ele foi escolhido por seguir o modelo de infraestrutura como serviço e, assim, permitir o controle integrado dos recursos físicos de computação, armazenamento e rede. Trata-se de um projeto aberto, desenvolvido por uma comunidade global e que conta com o apoio de organizações consolidadas no ramo de computação em nuvem. Essas carac-

terísticas colaboram para que seja uma plataforma em constante evolução que agregue as tendências do mercado. Além disso, é possível realizar instalações customizadas, selecionando os módulos que oferecem os serviços desejados [del Castillo et al. 2013]. As aplicações podem gerir os serviços da nuvem através das APIs do OpenStack, o que proporciona flexibilidade. Como plataforma para o controle dos recursos de rede, selecionamos o controlador Ryu, que pode receber comandos via API *RESTful*. Na camada física, serão utilizados hardware *commodity*, graças à flexibilidade oferecida pelos controladores de recursos escolhidos. Para atender ao requisito de oferecer uma ferramenta para controle de experimento, selecionamos o NEPI por ser compatível com a linguagem Python (usada no desenvolvimento do CMF). Assim, qualquer usuário com conhecimento de lógica de programação e da linguagem Python poderá criar *scripts* para a execução de experimentos com os recursos do *testbed*. Para atender ao requisito de monitoramento do *testbed*, selecionamos o Zabbix, por sua robustez.

4. Conclusão

A proposta apresentada diminui a lacuna entre o ambiente onde são feitas as provas de conceito e onde as soluções serão implantadas. A arquitetura proposta preza pela flexibilidade e baixo custo, adotando tecnologias abertas. O *Orquestrador* permite que o CMF suporte com eficiência experimentos que necessitam de migração e/ou escalada de recursos. O desenvolvimento do CMF do *testbed* FUTEBOL na UFES é um trabalho em andamento. Até a escrita desse trabalho, uma versão inicial do AM da nuvem e do *driver* para OpenStack já haviam sido desenvolvidos. Ambos estão disponíveis[5] e contam com instruções para instalação. Há uma demonstração em vídeo[6] desses componentes trabalhando na alocação de máquinas virtuais. O processo de definição dos requisitos do *testbed* se mostrou incremental e centrado nos aspectos de interoperabilidade, escalabilidade e código aberto.

Agradecimentos

Este trabalho tem recebido financiamento do projeto Horizon 2020 (União Européia) sob no. 688941 (FUTEBOL), assim como do MCTI por meio da RNP e do CTIC. Além disso, gostaríamos de agradecer o financiamento do CNPq/CAPES e FAPES.

Referências

Berman, M. et al. (2015). Future internets escape the simulator. *Communications of the ACM*, 58(6):78–89.

del Castillo, J. A. L., Mallichan, K., and Al-Hazmi, Y. (2013). Openstack federation in experimentation multi-cloud testbeds. In *Cloud Computing Technology and Science (CloudCom), 5th International Conference on*, volume 2, pages 51–56. IEEE.

Kim, H. and Lee, S. (2012). First cloud aggregate manager development over first: Future internet testbed. In *Information Networking (ICOIN), 2012 International Conference on*, pages 539–544. IEEE.

Sitaram, D., Harwalkar, S., and Kumar, K. S. (2016). Standards based integration of intercloud for federation with openstack. In *Cloud Computing in Emerging Markets (CCEM), 2016 IEEE International Conference on*, pages 113–118. IEEE.

[5]https://github.com/nerds-ufes/cmf-futebol

[6]https://goo.gl/D9OLn6

Mitigating the Risks of Supporting Multiple Control Planes in a Production SDN Network: A Use Case

Jeronimo Bezerra[1], Julio Ibarra[1], Marcos Schwarz[2], Humberto Freitas[2], Heidi Morgan[3]

[1]Florida International University – Miami, FL – USA

[2]Rede Nacional de Ensino e Pesquisa – Campinas, SP – Brazil

[3]University of Southern California – Los Angeles, CA – USA

{jbezerra, julio}@fiu.edu, {marcos.schwarz, humberto.galiza}@rnp.br, hlmorgan@isi.edu

Abstract. *The SDN paradigm enables network operators to host multiple control planes in parallel, being an approach to support multiple network services. Supporting multiple control planes over production networks exposes the production environment to potential risks and increases operational complexity. To understand and mitigate these risks, we implemented procedures and tools that resulted in a more reliable network. This paper describes our experience and findings with the support of multiple control planes in a wide-area production network.*

1. Introduction

Hosting multiple control planes enables SDN networks to offer specialized network services. However, supporting multiple control planes in a production SDN network involves potential risks and increases the complexity of operation and troubleshooting processes. Risks result from code instability in the OpenFlow agents, and the complexity of operation and troubleshooting is a consequence of extra protection layers and procedures required to handle the software instability.

With the vast adoption of OpenFlow [McKeown et al. 2008], network operators are being exposed to some operational gaps, such as the lack of specialized OpenFlow troubleshooting tools. AmLight has hosted multiple SDN control planes in parallel with production applications since 2014 [Ibarra et al. 2015] using the Flow Space Firewall (FSFW)[1], an OpenFlow proxy developed to support multiple control planes operating in parallel in an OpenFlow network. During this period, unexpected network outages were seen. Aiming to handle these operational OpenFlow gaps and increase the network's resilience, some procedures and tools were created.

In the first year after the OpenFlow activation, AmLight's production network was involved in nearly twenty network outages due to OpenFlow agents' crashes. In these situations, troubleshooting has proved to be difficult. In each event, OpenFlow switches and SDN applications' event logs and packet inspection were used. Nevertheless, unfortunately, the tools did not provide enough information, making it impossible to understand the event completely.

[1] "FSFW: Flow Space Firewall": http://globalnoc.iu.edu/sdn/fsfw.html

This paper's contribution is to describe the AmLight experience when hosting multiple control planes alongside with production applications and present some innovations developed to handle these operational gaps introduced with the OpenFlow deployment.

2. Innovations Developed to Minimize Risks on SDN Networks

Three main innovations were created to help to minimize the risks of supporting multiple SDN control planes: (1) an evaluation methodology, (2) an OpenFlow packet dissector (the OpenFlow Sniffer) and (3) an OpenFlow packet filter (the Testbed Sanitizer). These innovations are described in the next sections.

2.1. The Evaluation Methodology

Before hosting any new control plane, an evaluation methodology is performed. The evaluation methodology focuses on identifying OpenFlow messages that could affect the network resilience. The validation process identifies the OpenFlow messages used by the new control plane software using the OpenFlow Sniffer and validates these messages against the ones supported by the production OpenFlow switches. Then, the application is hosted in a testing environment with physical devices and, in case no impact is observed, a production slice is created for it.

2.2. The OpenFlow Sniffer

The OpenFlow Sniffer [Bezerra et al. 2016] is a tool developed with the focus on troubleshooting OpenFlow environments hosting multiple control planes. The OpenFlow Sniffer was developed to handle OpenFlow proxies, allowing network operators to associate a controller with an OpenFlow switch. The proxy support is especially useful in cases of asynchronous OpenFlow messages because only one type of OpenFlow message carries the *datapath-id* of the OpenFlow switch. Currently, the OpenFlow Sniffer is the only production tool available with such support, and it operates passively to lower the risks to the SDN controller's environment.

2.3. Testbed Sanitizer

Testbed Sanitizer is a tool created to facilitate the troubleshooting procedure and, at the same time, reduce exposure to risks. Its primary purpose is to filter all undesired OpenFlow messages per network device's line card and software version.

Filtering all undesired OpenFlow messages requires a catalog of all line cards and software versions in use. The OFTest[2] tool, a framework and test suite to validate compliance with the OpenFlow specification, was employed to create this catalog. After running OFTest's tests against every switch's line card, the output is parsed and exported to an XML file.

The Testbed Sanitizer works intercepting OFPT_FLOW_MOD messages received from all OpenFlow control planes and validating these messages against the catalog created with the OFTest. Only the OpenFlow messages added to the catalog are forwarded to the switches. Others are rejected with an OFP_ERROR message, thus, protecting the switches from unsupported OpenFlow messages.

[2] OFTest, Available: http://www.projectfloodlight.org/oftest/

3. AmLight Innovations Initiatives Evaluated

During the last two years, a few different SDN control planes were hosted at AmLight in parallel with production control planes. This section describes our experience with two of them: (1) the ONOS/SDN-IP application and (2) the FIBRE testbed.

3.1. Using ONOS as a use case

Open Network Operating System[3] (ONOS) is an open source OpenFlow controller developed with the focus on Internet Service Providers. ONOS has an application to handle BGP feeds, IP and IPv6 forwarding, called SDN-IP.

During the ONOS SDN-IP's evaluation process, AmLight engineers were not aware of the lack of support for MAC rewriting in one of AmLight's OpenFlow switches. Consequently, OpenFlow error messages were sent back and, as the FSFW proxy sits between the controller and OpenFlow switches, the process of locating the OpenFlow switch generating the error message was quite complex.

With the OpenFlow Sniffer's proxy support, different from any other traditional sniffer available, determining the OpenFlow switch without the MAC rewriting support became possible to be done in near real time. Similar troubleshooting methodology was utilized to identify modules that did not support matches based on TCP ports. With the OpenFlow Sniffer, all possible matches and actions used by ONOS/SDN-IP were mapped during the evaluation methodology process.

3.2. Using FIBRE as a use case

The FIBRE [Sallent 2012] federated research testbed has a set of wireless and OpenFlow devices available for experimentation. An overlay network (FIBREnet) interconnects all the facilities and enables wide-area OpenFlow experiments. FlowVisor [Sherwood et al. 2009] is being used as an SDN hypervisor to enable researchers to create network slices.

During FIBRE's evaluation process, the following challenges were found: (1) FlowVisor expected to fully control the OpenFlow switches, not a slice; (2) Use of untagged VLAN is hardcoded into FlowVisor but at AmLight, untagged VLAN was reserved for internal use; (3) FIBRE assumes that any OpenFlow controller can be used by the user but AmLight requires that all controllers needs to be validated through the evaluation methodology; (4) All OpenFlow features are provided to the FIBRE user. But, at AmLight, only risk-free features are allowed.

The challenges detected during the evaluation process forced AmLight to try a new approach: a new security layer was created to filter these unsupported OpenFlow messages. The Testbed Sanitizer was then created and was described in Section 2.3.

6. Findings

Most of the issues threatening network availability were stateful, not stateless. Stateful issues occur as a result of multiple messages, or a sequence of messages in a particular context. Even with the evaluation methodology in place, stateful issues may pass undetected. As a proof of concept, the Testbed Sanitizer has proved to be an interesting

[3] Open Network Operating System, Available: http://onosproject.org

approach to handling unsupported OpenFlow messages. However, it may require a significant development effort to address stateful issues. As a lesson learned, it became evident that we should work with the network device's vendor to improve its OpenFlow agent instead of investing in external security filters. Any extra layer of protection, in the end, also increases the operational complexity and should be avoided.

Additionally, the application of the evaluation methodology when starting a new control plane proved to be fundamental. Throughout the evaluation process, AmLight engineers could understand how the application worked and then be prepared for future troubleshooting processes. Even though the evaluation process is very useful, it is very time-consuming and needs to be automated for future evaluations.

Before having the Testbed Sanitizer and the OpenFlow Sniffer, troubleshooting activities used to last up to 30 hours, and, during the process, outages compromised the production network. With the innovations in place, all OpenFlow messages are traced effectively, and non-compliant OpenFlow messages are discarded in real-time. The number of outages that resulted from these stateless non-compliant OpenFlow messages dropped substantially: from 15 network outages to 0 in the first year.

7. Conclusions

Hosting multiple control planes in parallel has proven to be complex, but possible, manageable and beneficial to network operators. It requires a deep understanding of how network devices and protocols work, how to debug issues, and how to avoid impacts to the network resilience. Troubleshooting tools and OpenFlow agents still need to evolve to protect against a single experimental application compromising the overall availability of a production environment. The Evaluation Methodology, the OpenFlow Sniffer, and the Testbed Sanitizer have considerably reduced the potential risks of supporting parallel control planes at AmLight.

References

Bezerra, J., Galiza, H., Ibarra, J., & Schwarz, M. (2016) "AmLight's OpenFlow Sniffer dissected: Troubleshooting production networks". In Anais do WPEIF 2016 Workshop de Pesquisa Experimental da Internet do Futuro (p. 33). (to appear in proceedings).

Ibarra, J., Bezerra, J., Morgan, H., Lopez, L., Cox, D., Stanton, M., Machado, I. & Grizendi, E. (2015). "Benefits brought by the use of OpenFlow/SDN on the AmLight intercontinental research and education network". In 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM) (pp. 942-947). IEEE.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Turner, J. (2008). OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 38(2), 69-74.

Sallent, S., Abelém, A., Machado, I., Bergesio, L., Fdida, S., Rezende, J., Tassiulas, L. (2012). FIBRE project: Brazil and Europe unite forces and testbeds for the Internet of the future. In International Conference on Testbeds and Research Infrastructures (pp. 372-372). Springer Berlin Heidelberg.

Sherwood, R., Gibb, G., Yap, K. K., Appenzeller, G., Casado, M., McKeown, N., & Parulkar, G. (2009). Flowvisor: A network virtualization layer. OpenFlow Switch Consortium, Tech. Rep, 1-13.

A Simple Solution for IoT Experimentation in the Context of Future Internet Architectures

Ramon P. S. Chaib[1], **Antonio M. Alberti**[1]

[1]ICT Laboratory, Instituto Nacional de Telecomunicações
INATEL - João de Camargo 510, Centro, Santa Rita do Sapucaí
CEP 37540-000, Minas Gerais, Brazil. Phone: +55 35 3471 9218

ramonp@gec.inatel.br, alberti@inatel.br

Abstract. *In the last decade, many approaches appeared to revolutionize Internet architecture from scratch. They are collectively called Future Internet Architectures. In this paper, we address the challenge of experimenting with Internet of things in this context. Using open-source tools, we developed a standard scenario capable to evaluate proposals in a small scale first (locally, in laboratory), and latter in large scale cloud-based testing platform. As a use case of the proposal, we discuss its application to the eXpressive Internet Architecture.*

1. Introduction

This work proposes a solution for testing IoT over future Internet architectures (FIAs) [1]. The aim is to test how the architecture behaves while interoperating with motes running Contiki [2], an open source operating system for the Internet of Things. Using Cooja, a network simulator that allows networks of Contiki motes to be simulated even in a hardware level [2], and Docker [3], a software container platform, we created a standard evaluation scenario to compare distinct Internet architectures with minor implementations of components for each of them. Unlike virtual machines (VMs), containers do not bundle a full operating system, they initialize only libraries and settings required to make the software work as needed. This enables efficient, lightweight, self-contained systems that guarantee software will always run the same, regardless of where it is being deployed [3].

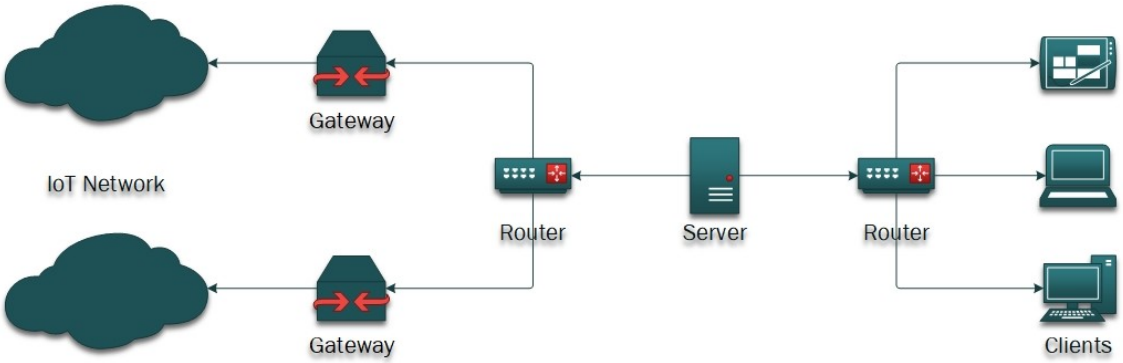
The first architecture to be tested is *XIA* (eXpressive Internet Architecture), an Future Internet Architecture with native support for multiple principals and the ability to evolve its functionality to accommodate new, as yet unforeseen, principals over time [4]. It will be testing interoperating with a 6LoWPAN IoT Network [5]. 6LoWPAN defines encapsulation and header compression mechanisms that allow IPv6 packets to be sent and received over LoWPANs. Specified by the IEEE 802.15.4 standard, LoWPAN is a simple low cost communication network that allows wireless connectivity in applications with limited power and relaxed throughput requirements [5].

The remainder of this article is organized as follows. Section 2 presents our system works and what we have to adapt for testing each architecture. Section 3 presents a study case where *XIA* [4] will be tested interoperating with a 6LoWPAN [5] IoT Network. Finally, Section 4 presents the final considerations, emphasizing both the main contributions of the article and future work.

2. Proposed Solution

Our system consist in a shell *script* that load five different types of *Docker Images* (*Client, Gateway, Client Router, Gateway Router, and Server*) connected in a star topol-

Figure 1. Proposed solution to evaluate IoT in future Internet architectures.



ogy. *Clients* are connected to the *Client Routers*, *Gateways* are connected to the *Gateway Routers*, *Client Routers* and *Gateway Routers* are connect to the *Server*.

There is only **one** *Server*, each *Server* can have **GR** *Gateway Routers* and **CR** *Client Routers*. Each *Gateway Router* can have **G** *Gateways*, while each *Client Router* can have **C** *Clients*. *Gateways* are running Cooja with **M** IoT motes, forming the *IoT Network*. **C CR G GR M** are input parameters of the *script*. The Figure 1 shows an example of the *script* configuration where the server is connected to **one** *Gateway Router* with **two** *Gateways* and **one** *Client Router* with **three** Clients. Each Gateway is simulating a *IoT Network*.

2.1. Echo Client/Server

The firmware of the motes is an *echo-server*, the *echo-server* receives data and replies the same data to the source. Each *Client* runs an *echo-client* that sends data to a mote randomly selected from a table previously stored in the *Server* and get its reply. The *echo-client* also make all the measures (delay, efficiency, load, errors, etc.). Everything will be stored and sent to the server at the end of the execution. Data size **S** and the number of requests **RQ** of each *Client* are also input parameters from the *script*.

2.2. Gateway Service

There is a *Gateway Service* running in the *Gateways*, the *Gateway Service* translate data from the *IoT Network* to the tested architecture. *echo-clients* and *echo-servers* will talk directly to it. In the initialization step, the *Gateway Service* store all motes addresses and send to a table in the *Server*.

2.3. Docker Images

All *Docker Images* must have the tested architecture running with all necessary procedures for its operation configured in the initialization step. *Docker Images* are started in the following order: *Server, Gateway Routers, Client Routers, Gateways, Clients*.

- *Server*: Configured as a router of the tested architecture. Connects all routers, store test results and address tables.
- *Client Routers/Gateway Routers*: Configured as routers of the tested architecture.
- *Gateways*: Configured as hosts of the tested architecture, starts Cooja and the *Gateway Service* in the initialization step.
- *Clients*: Configured as hosts of the tested architecture, starts *echo-client* on the initialization step.

3. Study Case: XIA / 6LoWPAN

In the first test, a *6LoWPAN* service will be used by a purely *XIA* application. For this, *IPv6* addresses need to be bound to *XIA* identifiers (*XID*) [4]. The *XIA* architecture defines several *XIA* identifier types with distinct semantics of communication, processing that is required to forward packets, and intrinsic security properties [4]. In this case, a *SID* (Service Identifier) type will be used. *SIDs* support communication with services [4].

3.1. XIA Docker Images

All *Docker Images* will be running *XIA Prototype* [4] in Ubuntu Linux containers. *XIA Prototype* allows users to Run Sample Applications over the *XIA* network and write *XIA* applications. The prototype is implemented on top of the *click modular router* [6], which will create some overhead and increase the *Docker Images* complexity.

- *XIA Server*: First image to be initialized. It is configured as an *XIA router*. It starts with an empty address table that will be populated by the *Gateway Services* later.
- *XIA Gateway Routers*: Will be initialized after the *XIA Server* has been fully started. They are configured as *XIA routers* and their only function is to route packets between *XIA Gateways* and the *XIA Server*.
- *XIA Client Routers*: Will be initialized after the *XIA Server* has been fully started. They are configured as *XIA routers* and their only function is to route packets between *XIA Clients* and the *XIA Server*.
- *XIA Gateways*: Will be initialized after all routers have been fully started. Configured as *XIA hosts*. First they start Cooja and then the *Gateway Service*.
- *XIA Clients*: The last images to be initialized. They copy the address table stored on the *XIA Server* and then initialize the echo-client.

3.2. 6LoWPAN echo-server/Cooja

A simple 6LoWPAN echo-server that receives a package and replies the same to the source. It will be running in the Cooja motes over a simulated 6LoWPAN IoT Network.

3.3. XIA/6LoWPAN Gateway Service

The *Gateway Service* will list all IPv6 addresses of the motes and assign a *SID* to each one creating a table. Another table containing only the *SIDs* will be concatenated to the address table on the *Server*. After that the *Gateway Service* stays alive waiting to route packets between *echo-clients* and *echo-servers*. When a packet is forwarded to a *echo-server*, the *Gateway Service* waits for the response to forward back to the source *echo-client* until a preset timeout. Any packet to the same *echo-server* received during this period will be buffered until it can be forwarded. If a timeout happens the *Gateway Service* replies to the *echo-client* an error message indicating packet loss on the *IoT Network*.

3.4. XIA Echo Client

The *echo-client* generate the messages and send to the *echo-servers* (*SID*) randomly chosen from the address table. For each generated message, the *echo-client* waits for its response, measuring: latency, packets lost in the *XIA network* (determined by a timeout), packets lost in the IoT network (determined by the Gateway Service error message) and errors (corrupted packets). Everything will be stored in a log and sent to the server at the end of execution for analysis.

4. Concluding Remarks

Once a test scenario is developed, running the tests will require just a single command in the terminal or minor implementations to test other proposals. *echo-client* and *Gateway Service* need to adapted using the new architecture APIs. *Docker Images* need to be adapted as well, but they are easy to prepare and similar to one another. The *echo-server* and the *script* remains the same. When a small test is successful in a single computer, we can move for larger tests using computational clouds just choosing new values for the topology. By employing open-source tools Docker and Cooja, this solution has very low cost and covers all network overheads since its components are actually implemented. Other applications can be developed as long as it is possible to implement a gateway capable of translating the messages among the architectures.

As a future work, we will be testing *Linux XIA* [7], a native implementation of *XIA* in the Linux kernel. Unlike *XIA Prototype*, *Linux XIA* does not use the *click modular router*, which will reduce the overhead and facilitate the preparation of the *Docker Images*.

References

[1] P. Stuckmann and R. Zimmermann, "European research on future internet design," *IEEE Wireless Communications*, vol. 16, no. 5, pp. 14–22, October 2009.

[2] Contiki. (2017, Mar.) Contiki website. [Online]. Available: http://www.contiki-os.org/

[3] Docker. (2017, Mar.) Docker website. [Online]. Available: https://www.docker.com/

[4] D. Han, A. Anand, F. Dogar, B. Li, H. Lim, M. Machado, A. Mukundan, W. Wu, A. Akella, D. G. Andersen, J. W. Byers, S. Seshan, and P. Steenkiste, "Xia: Efficient support for evolvable internetworking," *NSDI'12, USENIX Association*, pp. 23–23, 2012.

[5] G. Montenegro, C. Schumacher, and N. Kushalnagar, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals," RFC 4919, Aug. 2007.

[6] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[7] M. Machado, C. Doucette, and J. W. Byers, "Linux xia: An interoperable meta network architecture to crowdsource the future internet," *ANCS '15, IEEE Computer Society*, pp. 147–158, 2015.

Acknowledgments

This work was partially supported by Finep, with resources from Funttel, Grant No. 01.14.0231.00, under the Radiocommunication Reference Center (Centro de Referência em Radiocomunicações- CRR) project of the National Institute of Telecommunications (Instituto Nacional de Telecomunicações - Inatel), Brazil. The authors also would like to thank CNPq, CAPES, FINATEL and FAPEMIG.

Autenticação e Controle de Acesso na Arquitetura ETArch

Pedro H. A. Damaso Melo[1], **Flávio de O. Silva**[1], **Pedro F. Rosa**[1]

[1]Faculdade de Computação (FACOM)
Universidade Federal de Uberlândia (UFU) – Uberlândia, MG – Brasil

{pedro.damaso,flavio, pfrosi}@ufu.br

Resumo. *Apesar das evoluções, a Internet atual não consegue tratar adequadamente requisitos como* multihoming*, QoS, mobilidade,* multicast *e segurança. Vários grupos de pesquisa ao redor mundo estão envolvidos em criar, de forma experimental e incremental, a próxima geração da arquitetura da Internet. Uma inciativa brasileira nessa área é a* Entity Title Architecture *(ETArch) cujo protótipo é baseado no conceito de* Software Defined Networking *e utiliza o protocolo OpenFlow. Este trabalho apresenta duas contribuições implementadas na ETArch, sendo um mecanismo de autenticação e um de controle de acesso, cujas análises de custo-benefício são apresentadas neste artigo.*

Abstract. *Despite of evolutions, the current Internet can not adequately handle requirements such as multihoming, QoS, mobility, multicast and security. Several research groups around the world are involved in creating, experimentally and incrementally, the next generation of Internet architecture. A Brazilian initiative in this area is the Entity Title Architecture (ETArch) whose prototype is based on the concept of Software Defined Networking and uses the OpenFlow protocol. This work presents two contributions implemented in ETArch, being an authentication mechanism and an access control, whose trade-off is presented in this work.*

1. Introdução

Os avanços tecnológicos tanto em hardware quanto em software, as novas tecnologias de acesso em banda larga e as redes de telecomunicações móveis propiciaram o surgimento de novos serviços e aplicações que seriam difíceis de se imaginar nos anos setenta quando os protocolos da internet foram especificados. Mesmo com evoluções, a Internet atual não consegue tratar adequadamente requisitos como *multihoming*, QoS, mobilidade, *multicast* e segurança [Handley 2006]. Vários grupos de pesquisa ao redor mundo estão envolvidos em criar a próxima geração da arquitetura de Internet [Pan et al. 2011].

No que tange às novas arquiteturas, o Brasil possui algumas iniciativas, sendo uma delas a *Entity Title Architecture* (ETArch) [Guimaraes et al. 2014]. Ela possui uma visão conceitual muito próxima da abstração proposta pelas Redes Definidas por Software e, portanto, desde o seu primeiro protótipo utiliza o protocolo OpenFlow para materializar essa visão. Desde a sua criação, pesquisadores de várias universidades vêm trabalhando para incorporar à ETArch, de forma incremental, soluções que visam atender os requisitos de Internet do Futuro.

Na ETArch o endereçamento e a identificação são baseados em uma designação independente da topologia que identifica de forma única uma entidade, chamada Título.

A comunicação entre múltiplas entidades ocorre através de barramento lógico, chamado *Workspace*.

Um componente central da ETArch é o *Domain Title Service* (DTS) que representa o plano de controle da rede. O DTS é composto de *Domain Title Service Agents* (DTSAs) que mantém informações sobre as entidades registradas no domínio e os *Workspaces* em que as mesmas estão anexadas. O DTSA é responsável por configurar os elementos de rede para permitir as entidades associadas a um dado *Workspace* possam participar da comunicação.

Apesar dos diversos incrementos projetados, implementados e agregados à ETArch, nenhum deles se relaciona aos aspectos da segurança. Sendo assim, as principais contribuições deste trabalho são elaborar e implementar dois mecanismos: um para autenticação; e outro para o controle de acesso ao ambiente de comunicação oferecido pela ETArch.

Para validar os mecanismos de autenticação e controle de acesso, foi realizada uma avaliação experimental, que objetiva demonstrar os benefícios dos mecanismos criados para a arquitetura. O cenário de testes apresentado demonstra a viabilidade e a importância deste trabalho.

Este trabalho está organizado da seguinte forma: A Seção 2 apresenta a visão geral dos mecanismos de segurança. A Seção 3 descreve o cenário utilizado para avaliação experimental e apresenta os resultados obtidos. Finalmente, a Seção 4 apresenta as conclusões e trabalhos futuros.

2. Proposta

O surgimento de SDN na área de redes de computadores modifica a maneira como alguns aspectos da comunicação de dados podem ser tratados. O SDN impõe que alguns aspectos relacionados à segurança seja tratado no plano de controle, por exemplo, no momento em que uma entidade solicita o seu registro na arquitetura ETArch. O presente trabalho objetiva demonstrar experimentalmente o uso do plano de controle para fornecer mecanismos de autenticação e controle de acesso.

Na arquitetura ETArch, a autenticação de uma entidade é realizada no momento em que a entidade se registra na rede e o controle de acesso é realizado no momento em que uma entidade tenta se conectar a um *Workspace*, ou seja, quando deseja participar de um domínio de comunicação.

O mecanismo de autenticação foi desenvolvido tendo como ponto de partida a recomendação X.811 [UNION 1995a], porém considerando a filosofia do paradigma SDN e as particularidades da arquitetura. Na ETArch, o conceito de entidade é genérico e pode ser entendido como tudo que possui capacidade de se comunicar, sendo assim, a autenticação é tratada de uma forma onde é possível adaptar-se a diversos cenários. Por exemplo, esse mecanismo precisa autenticar aplicações, elementos de rede, sensores, *smartphones*, usuários, entre outros.

O controle de acesso se baseou na recomendação X.812 [UNION 1995b], e é aplicado ao plano de controle da arquitetura ETArch para verificar quais entidades possuem privilégio para fazer parte de determinado *Workspace*. A *Access Control Enforcement Function* (AEF) é uma função especializada que faz parte do caminho de acesso entre as

entidades e um *Workspace* em cada solicitação de acesso e reforça a decisão tomada pela *Access Control Decision Function* (ADF).

3. Avaliação Experimental e Análise de Resultados

Nesta seção são avaliados os mecanismos de autenticação e controle de acesso propostos tendo-se como cenário uma aplicação de $chat$. Inicialmente descreve-se o cenário usado na prova de conceito e, posteriormente, uma análise dos resultados obtidos.

3.1. Cenário Experimental

Para o DTSA foi utilizado um computador com o sistema operacional Ubuntu 16.04.1 LTS, com o processador Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz e 6 GB de memória RAM. Foi utilizado o Mininet para simular uma topologia de rede usando uma maquina virtual com sistema operacional Ubuntu 14.04 LTS com 512MB de memória RAM. Foi criada uma topologia com *OpenFlow Switches* representados pelo conjunto $\{$s1, s2, s3, s4, s5, s6$\}$ e 15 hosts $\{$h1, h2, h3, h4, h5, h6, h7, h8, h9, h10, h11, h12, h13, h14, h15$\}$.

Para o teste, foi utilizado uma aplicação de $chat$, onde foi criado um *Workspace* chamado $W1$ e os *hosts* representados na topologia se conectaram a esse $W1$, sendo assim, foi medido o tempo que os *hosts* levaram para realizar o seu registro no ETArch (primitiva *Entity Register*) e o tempo para realizar o *Workspace Attach*, ou seja, para conectar-se ao *Workspace*. Cada experimento, envolvendo toda a topologia, foi executado 10 vezes.

3.2. Resultados Obtidos

Antes dessa implementação a ETArch não dispunha de mecanismo de segurança, o que pode causar diversos problemas, sendo que após essa proposta é possível realizar a autenticação e o controle de acesso de entidades na arquitetura ETArch. Para os testes foram utilizados certificados digitais próprios para cada *host*.

A Figura 1 apresenta dois gráficos: à esquerda, os tempos auferidos para o serviço de autenticação *Entity Register*; à direita, os tempos do serviço de controle de acesso *Workspace Attach*, considerando dois cenários, com (azul) e sem (vermelho) o uso dos mecanismos de segurança. No *Entity Register*, o tempo médio com segurança foi de 28.08 ± 2.67 ms enquanto que o tempo médio sem segurança foi de 17.64 ± 4.05 ms. Isso representa um acréscimo médio de 10.45ms no cenário com segurança. Para o *Workspace Attach*, o tempo médio foi de $28.47 \pm 6.33ms$ e $19.10 \pm 4.1ms$ nos cenários com e sem segurança respectivamente, indicando um acréscimo médio de 9.37 ms. Os valores médios são mostrados com um intervalo de confiança de 95% utilizando uma distribuição T-Student.

O acréscimo médio é comparativamente pequeno em termos de tempo se se considerar os benefícios dos mecanismos de autenticação e controle de acesso incorporados à ETArch.

4. Conclusões e Trabalhos Futuros

Neste trabalho foi apresentada a inclusão de mecanismos na arquitetura ETArch com o objetivo de realizar a autenticação e o controle de acesso de entidades no plano de controle. O trabalho mostrou o custo-benefício dos mecanismos acrescentados. Apesar do

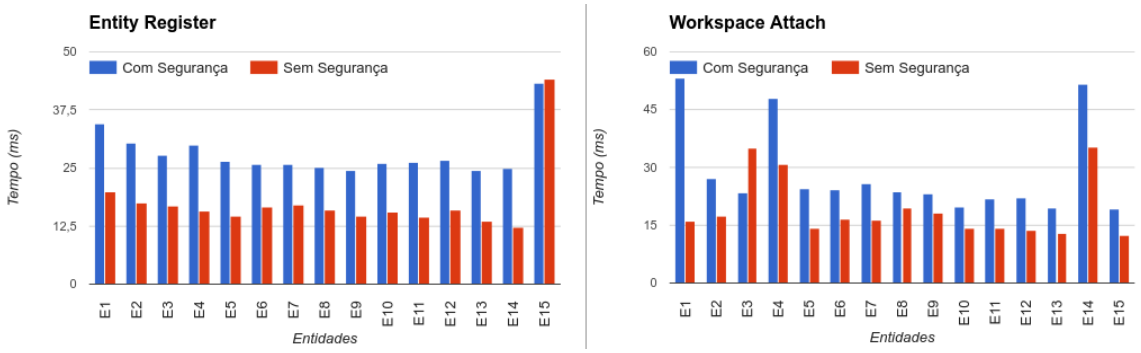


Figura 1. Avaliação Comparativa para o *Entity Register* e o *Workspace Attach*

acréscimo médio de tempo ser de 10.45ms para o *Entity Register* e 9.37ms no *Workspace Attach* há um ganho indiscutível em relação à segurança com os mecanismos de autenticação e autorização desenvolvidos.

Aspectos tais como detecção de intrusos e confidencialidade serão tratados pela arquitetura futuramente. Considerando que os DTSAs (agentes do plano de controle) mantêm uma relação de confiança, então esse processo é realizado uma vez. Em caso de mobilidade, por exemplo, quando a entidade troca de localidade, esse outro DTSA se encarregará de obter essas informações de acesso no DTSA de origem, onde foi feito o registro da entidade.

5. Agradecimento

Esse trabalho foi financiado pela CAPES atraves do Programa de Apoio ao Ensino e à Pesquisa Científica e Tecnológica em Defesa Nacional (Pró-Defesa).

Referências

[Guimaraes et al. 2014] Guimaraes, C., Corujo, D., Silva, F., Frosi, P., Neto, A., and Aguiar, R. (2014). IEEE 802.21-enabled Entity Title Architecture for handover optimization. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2671–2676.

[Handley 2006] Handley, M. (2006). Why the Internet Only Just Works. *BT Technology Journal*, 24(3):119–129.

[Pan et al. 2011] Pan, J., Paul, S., and Jain, R. (2011). A survey of the research on future internet architectures. *IEEE Communications Magazine*, 49(7):26–36.

[UNION 1995a] UNION, I. T. (1995a). X.811:Information technology - Open Systems Interconnection - Security frameworks for open systems: Authentication framework.

[UNION 1995b] UNION, I. T. (1995b). X.812:Information technology - Open Systems Interconnection - Security frameworks for open systems: Access control framework.

Realização

Apoio Fomento

Apoio Institucional

Patrocinador Diamante

Núcleo de Informação e Coordenação do Ponto BR

Comitê Gestor da Internet no Brasil

Patrocinador Ouro

Patrocinador Bronze