Implementação de um switch em P4 com suporte simultâneo a múltiplas arquiteturas de Internet do Futuro

José Augusto T. Gavazza[1], Michael dos Santos[1], Paulo Ditarso Maciel Jr.[1,2], Fábio Luciano Verdi[1], José A. S. Monteiro[3]

[1]Departamento de Computação (DComp) — UFSCar – Sorocaba

[2]Unidade Acadêmica de Informação e Comunicação — IFPB – João Pessoa

[3]Centro de Informática — UFPE — Recife

gavazza@gmail.com, michael-santos@outlook.com.br

paulo.maciel@ifpb.edu.br, verdi@ufscar.br, suruagy@cin.ufpe.br

Abstract. *New Future Internet Architectures have emerged independently in different locations, seeking to provide a new evolutionary environment for the Internet and to solve many issues that the current architecture, even with the creation of new protocols and patches, does not solve adequately. Despite efforts to deploy a single Internet architecture, what has been observed is the tendency to deploy and use different architectures that can coexist simultaneously. Therefore, there is an imminent need for a solution that makes this coexistence possible. The focus of this work is to present the implementation of a multi-architecture switch in P4 language, so that entities and applications from the same architecture can communicate.*

Resumo. *Novas arquiteturas de Internet do Futuro surgiram de forma independente em diferentes locais, buscando proporcionar um novo ambiente de evolução para a Internet e solucionar diversas questões que a arquitetura atual, mesmo com a criação de novos protocolos e patches, não resolve de maneira adequada. Apesar dos esforços em se implantar uma única arquitetura de Internet, o que se tem observado é a tendência de implantação e uso de diferentes arquiteturas que possam coexistir simultaneamente. Sendo assim, há a necessidade de uma solução que torne esta coexistência possível. O foco deste artigo é apresentar a implementação de um switch multiarquitetura em linguagem P4, para que entidades e aplicações de uma mesma arquitetura possam se comunicar.*

1. Introdução

Atualmente, existem diversas arquiteturas de rede com protótipos em diferentes estágios de implementação, cada qual com objetivos específicos de projeto e paradigmas de comunicação diferentes. Sendo assim, uma solução viável onde diferentes arquiteturas possam coexistir na mesma infraestrutura de rede é extremamente desejável.

O foco deste trabalho é implementar um switch usando a linguagem P4 capaz de distinguir diferentes arquiteturas e realizar, correta e simultaneamente, o encaminhamento

de pacotes entre entidades de mesma arquitetura de Internet. A opção pela implementação em linguagem P4 advém da possibilidade do switch ser executado em hardware programável e pela facilidade de expressar regras de encaminhamento em uma linguagem de alto nível.

O switch P4 suportará três arquiteturas de Internet: duas arquiteturas de Internet do Futuro originadas no Brasil, ETArch [de Oliveira Silva et al. 2012] e NovaGenesis [Alberti et al. 2017], além da tradicional arquitetura TCP/IP tendo em vista que esta arquitetura continuará existindo. Contudo, a solução do switch apresentado é flexível, podendo ser facilmente estendida para suportar novas arquiteturas de Internet do Futuro.

Assim como no trabalho apresentado em [Gupta et al. 2015], o switch aqui desenvolvido também pode ser utilizado em IXPs (Internet Exchange Points). Entretanto, o switch em P4 descrito neste artigo não é apenas uma proposta para um IXP tradicional (IP-only), mas sim para um novo modelo de IXP que possibilite o encaminhamento simultâneo de pacotes entre entidades de mesmas arquiteturas de Internet do Futuro. Considera-se este novo IXP como um ponto de troca de Internet do Futuro (FIXP, do inglês *Future Internet eXchange Point*).

2. Programming Protocol-independent Packet Processors – P4

P4 é uma linguagem de alto nível proposta a permitir a programação do plano de dados de dispositivos de rede e a programação de processadores de pacotes independentes de protocolo, que trabalha em conjunto com protocolos de controle de SDN (Software Defined Networking), permitindo criar soluções customizadas. Dentre os seus principais objetivos pode-se citar: (i) a reconfigurabilidade em tempo de execução, permitindo a programadores alterarem como os switches processam os pacotes depois de implementados; (ii) a independência do protocolo, não vinculando os elementos de rede a nenhum protocolo específico de redes; (iii) a independência do dispositivo alvo, permitindo que as funcionalidades de processamento de pacotes sejam independentes das especificidades do hardware alvo, através da simulação em ambiente virtualizado ou da execução em hardware programável [Bosshart et al. 2014].

Um programa implementado em P4 se utiliza da definição de cabeçalhos (*headers*), que descrevem a sequência e a estrutura de uma série de campos; de analisadores (*parsers*), que definem como os campos de cada cabeçalho são identificados e validados; de tabelas (*tables*), que são mecanismos utilizados para realizar o processamento dos pacotes; das ações (*actions*) que podem ser executadas de acordo com o processamento de pacotes; e, por fim, da implementação de programas de controle (*control programs*), que determinam a ordem das tabelas que são aplicadas a um pacote. Assim, a linguagem P4 possibilita a implementação de switches mais flexíveis, possibilitando ao programador decidir como o plano de dados processa os pacotes, sem se preocupar com detalhes de implementação.

3. Arquiteturas de Internet do Futuro

Nesta seção, são apresentadas e descritas as duas arquiteturas de Internet do Futuro inicialmente suportadas pelo switch.

3.1. ETArch – Entity Title Architecture

A ETArch [de Oliveira Silva et al. 2012] é uma Arquitetura de Internet do Futuro baseada no conceito SDN [Cox et al. 2017] e que apresenta como proposta resolver as novas demandas da Internet. Para isto, oferece suporte aos novos requisitos de aplicações como *multicast*, mobilidade, qualidade de serviço (QoS) e segurança; enquanto se mantém genérica o suficiente para suportar evoluções tecnológicas.

3.2. NG – NovaGenesis

NovaGenesis (NG) [Alberti et al. 2017] é uma Arquitetura de Internet do Futuro baseada em três pilares principais: (i) nomeação, resolução de nomes e cache de rede; (ii) dispositivos físicos, serviços e ciclos de vida de conteúdo; (iii) representantes das entidades para expor suas características e para fornecer operações definidas por software. NG visa criar uma arquitetura de informação convergente definida por um serviço que integra a troca, o armazenamento e o processamento de dados. Tal arquitetura também é escalável, concentrando-se em fornecer uma melhor coerência entre dados nomeados, interações baseadas em contratos entre entidades e controle, além de gerenciamento definido pelo serviço [Alberti et al. 2017].

4. Proposta da Solução e Implementação

Para a implementação do switch P4 proposto neste trabalho, foi utilizada a revisão de 2016 da linguagem de programação P4.

A codificação do switch P4 apresenta três seções funcionais como pode ser visto na Figura 1.

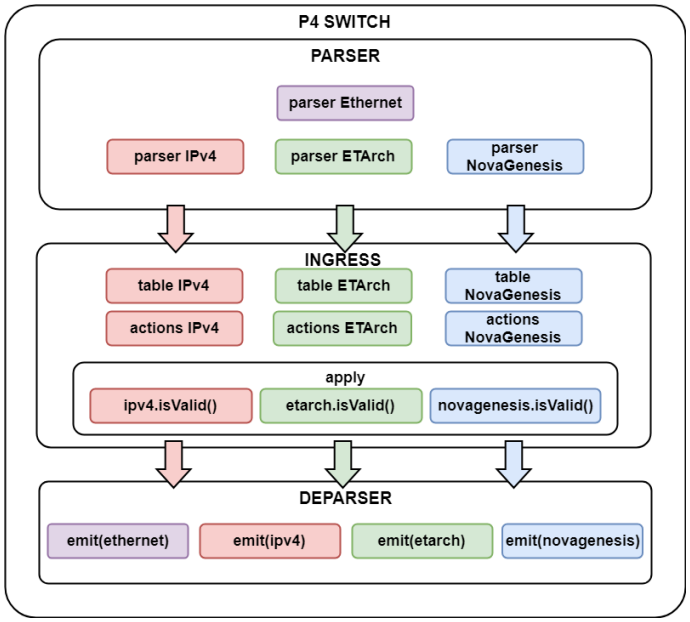

Figura 1. Representação da estrutura do código do switch P4.

A primeira seção corresponde ao analisador (*Parser*), que recebe os pacotes, extrai os cabeçalhos do protocolo Ethernet e de outros protocolos existentes e realiza a identificação da arquitetura do pacote recebido, através da análise do campo *ethertype* do

protocolo Ethernet. Na seção de ingresso (*Ingress*), são definidas as estruturas das tabelas de encaminhamento de pacotes, os campos-chave utilizados pelas regras de roteamento e a implementação das ações (*Actions*) de encaminhamento para cada uma das arquiteturas de Internet do Futuro implementadas. Por último, tem-se uma seção depuradora (*Deparser*), que remonta os campos dos cabeçalhos de cada protocolo no pacote a ser encaminhado à porta de saída correta do switch.

Embora não exista uma seção específica na linguagem P4 para a definição dos cabeçalhos das arquiteturas, os mesmos foram definidos no início do código P4 do switch, antes da codificação das principais seções funcionais. A definição das regras de encaminhamento de pacotes é configurada externamente, em arquivos individuais para cada arquitetura. A implementação foi realizada no ambiente Linux Ubuntu versão 16.10, utilizando a versão 2 do P4 *Behavioral Model* e o compilador *p4c*.

5. Prova de Conceito

A execução da prova de conceito foi realizada em um cenário virtualizado em ambiente Mininet. Foram configurados seis *hosts*, cada um com um endereço IPv4, um endereço MAC e uma *hash* de 6 bytes utilizada como identificador do título do *workspace* para a Etarch. Também foram configurados quatro switches FIXP, cada um com as regras de encaminhamento de pacotes para as arquiteturas IPv4, ETArch e NG. Dessa forma, cada dispositivo é compatível com essas arquiteturas, permitindo a cada *host* enviar e receber pacotes das três arquiteturas.

Utilizou-se a biblioteca Scapy[1] do Python para a geração e envio dos pacotes de testes, onde foram implementados os cabeçalhos das três arquiteturas abordadas. Foram criados dois programas, *send.py* e *receive.py*, que são executados nos terminais dos dispositivos. O *send.py* é utilizado para realizar a geração e envio de pacotes de uma determinada arquitetura definida pelo usuário, enquanto que o *receive.py* é utilizado para realizar o recebimento de pacotes enviados por qualquer uma das três arquiteturas definidas, independente do dispositivo que originou o pacote e também apresentar os dados do pacote recebido no terminal.

A Figura 2 ilustra a topologia utilizada para realização dos testes. Os dispositivos foram nomeados de *Host 1* a *Host 6* e a comunicação entre eles é realizada pelos quatro switches, de acordo com a figura. Cada *host* pode disparar pacotes das arquiteturas IPv4, ETArch ou NG. Nos testes realizados ficou definido que o *Host 1* envia pacotes IPv4 para o *Host 6*; o *Host 2* envia pacotes ETArch para o *Host 5*; e o *Host 3* envia pacotes NovaGenesis para o *Host 4*. A Figura 2 também apresenta os fluxos do encaminhamento dos pacotes durante os testes.

Os pacotes gerados em cada *Host* (1, 2 ou 3) são recebidos pela seção *Parser* do *Switch 1* e têm o campo Ethertype do protocolo Ethernet analisados quando chegam. Através do valor deste campo [2], o *Switch 1* identifica que o pacote pertence a uma das três arquiteturas (IPv4, ETarch ou NG). Em seguida, é invocada uma seção *Ingress* para o processamento da arquitetura correspondente, que compara o endereço de destino presente no cabeçalho do pacote e consulta as regras de encaminhamento da arquitetura correspondente, definidas na tabela de encaminhamento do *Switch 1*. Quando há correspondência

[1] https://scapy.net/.

[2] Valor 0x0800 para IPv4, valor 0x0880 para ETArch e valor 0x1234 para NG.

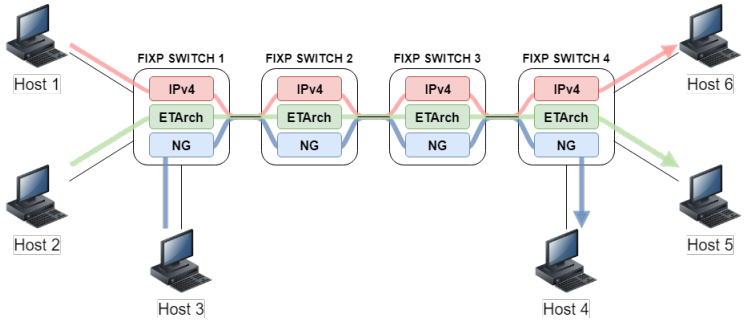

Figura 2. Encaminhamento dos pacotes durante a prova de conceito.

do destino com uma ocorrência na tabela de encaminhamento, o pacote é direcionado para a porta de saída que conecta o *Switch 1* ao *Switch 2* e o fluxo é finalizado pela seção *Deparser* que reconstitui o pacote na porta de saída determinada. Este procedimento é repetido até a chegada do pacote ao *Switch 4*, que realiza a análise do Ethertype e verifica o endereço de destino nas regras da tabela de encaminhamento, destinando assim o pacote à porta de saída onde se conecta o *Host* correspondente (6, 5 ou 4).

É importante observar que cada switch P4 atua de forma independente para cada arquitetura, proporcionando um dispositivo multiarquitetura, com suas próprias regras e tabelas de encaminhamento utilizadas de forma concorrente para cada arquitetura.

Foram executadas duas baterias de testes com o intuito de realizar uma análise inicial sobre o tempo de processamento de encaminhamento dos pacotes. As duas baterias de testes consistiram em: testes individuais para cada arquitetura e testes com as três arquiteturas funcionando simultaneamente. Em cada bateria, foram realizados experimentos com o envio de mil e dez mil pacotes, repetindo-se trinta vezes cada cenário. Os resultados do tempo médio de envio total dos pacotes (em milissegundos) podem ser observados na Tabela 1 (teste individuais) e Tabela 2 (testes simultâneos). Os intervalos de confiança para cada cenário foram obtidos com um nível de confiança de 95%.

Tabela 1. Dados do tempo médio de envio (ms) nos testes individuais.

Qtde pacotes	IPv4	ETArch	NG
1k	$7218,83 \pm 16,79$	$3638,27 \pm 27,25$	$3564,40 \pm 32,31$
10k	$72030,27 \pm 208,77$	$36258,57 \pm 154,74$	$34919,40 \pm 189,69$

Tabela 2. Dados do tempo médio de envio (ms) nos testes simultâneos.

Qtde pacotes	IPv4	ETArch	NG
1k	$21241,73 \pm 254,61$	$13114,33 \pm 183,52$	$12878,13 \pm 174,26$
10k	$179619,20 \pm 672,40$	$131786,50 \pm 583,93$	$129229,70 \pm 505,26$

Pelos dados apresentados em ambas as tabelas, nota-se que os tempos médios são estatisticamente diferentes para as arquiteturas avaliadas, com erro máximo de 1.4%, muito embora próximos quando consideradas a ETArch e NG. Pode-se observar que o IPv4 leva mais tempo para encaminhar os pacotes em comparação às demais arquiteturas. Este comportamento acontece provavelmente por uma maior inspeção de campos na arquitetura IPv4. Também é observado que, tanto individualmente quanto simultaneamente,

cada arquitetura mantém uma proporção (aparentemente linear) de crescimento de tempo da execução do cenário de mil para dez mil pacotes. Por fim, observa-se uma queda no desempenho do processo de encaminhamento de pacotes ao executar as três arquiteturas simultaneamente, quando comparados aos valores dos testes individuais.

Como resultado dos testes realizados no ambiente virtualizado, foi possível observar também que a atual implementação P4 do switch funcionou satisfatoriamente, realizando o encaminhamento de pacotes entre os dispositivos de mesma arquitetura, conforme esperado. Esta implementação cria um tratamento de pacotes dependente da arquitetura, com regras distintas de encaminhamento para cada uma das mesmas, agindo como um switch multiarquitetura.

6. Conclusão

Este trabalho apresenta a proposta e implementação de um switch multiarquiteturas programado na linguagem P4, que permite a interconexão simultânea de entidades de mesma arquitetura através da infraestrutura de rede existente. Como resultados dos testes de prova de conceito do switch P4, observa-se que o encaminhamento dos pacotes de três arquiteturas distintas de forma simultânea entre os dispositivos de mesma arquitetura foi realizado com sucesso. Como trabalho futuro, em uma segunda etapa deste trabalho, pretende-se estender a funcionalidade do switch P4 para que o mesmo possa realizar a comutação interna entre arquiteturas distintas.

Agradecimentos

Trabalho financiado pelo Projeto FAPESP “Alavancando a Internet do Futuro no Brasil através da Coexistência e Interconexão de Múltiplas Arquiteturas” (2015/24518-4). Agradecimentos também aos autores das arquiteturas ETArch e NovaGenesis pelas importantes discussões.

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.

Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., et al. (2014). P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95.

Cox, J. H., Chung, J., Donovan, S., Ivey, J., Clark, R. J., Riley, G., and Owen, H. L. (2017). Advancing Software-Defined Networks: A Survey. *IEEE Access*, 5:25487–25526.

de Oliveira Silva, F., Goncalves, M., de Souza Pereira, J., Pasquini, R., Rosa, P., and Kofuji, S. (2012). On the analysis of multicast traffic over the Entity Title Architecture. In *2012 18th IEEE International Conference on Networks (ICON)*, pages 30–35.

Gupta, A., Vanbever, L., Shahbaz, M., Donovan, S. P., Schlinker, B., Feamster, N., Rexford, J., Shenker, S., Clark, R., and Katz-Bassett, E. (2015). Sdx: A software defined internet exchange. *ACM SIGCOMM Computer Communication Review*, 44(4):551–562.