

프로그래밍 가능한 데이터 평면 기술 연구 동향

백상헌 고려대 전기전자공학부 / FIF Architecture Security WG

네트워크의 효율적 운영을 위해 소프트웨어 정의 네트워킹 (Software Defined Networking, SDN) 도입이 증가하고 있다. 이와 함께 데이터 평면에서도 소프트웨어 정의 네트워킹을 보다 유연하게 지원할 수 있도록, 프로그래밍 가능한 화이트 박스 스위치로 개발이 요구되고 있다. 이러한 프로그래밍 가능한 스위치를 통해 오픈플로우 (Openflow)의 버전 업데이트에도 유연하게 대응할 수 있고 새로운 서비스 기능이 출현하더라도 추가적인 장비 구매를 최소화시킬 수 있다. 프로그래밍 가능한 스위치 시스템을 개발하기 위해서는 기본적인 프로그래밍 가능 칩 개발과 개발자가 보다 쉽게 스위치 시스템을 재구성할 수 있는 새로운 데이터 평면 프로그래밍 언어가 필요하다. 기존의 프로그래밍 가능한 스위치 시스템에서도 일부 제한된 기능과 하드웨어/프로토콜에 종속적인 형태의 프로그래밍 언어가 존재하였는데 이러한 한계점을 극복하고자 하는 패킷 프로세싱 구현 언어인 P4 (Programming Protocol-Independent Packet Processors) [Bos14]가 많은 주목을 받고 있다.

P4는 오픈소스형태로 개발되고 있으며, 개발을 위해 화웨이, 시스코 등의 산업계 뿐만 아니라, 미국의 프린스턴 대학, 스탠포드 대학 등이 P4 컨소시엄에 참여하고 있다. P4는 기존의 FPGA와 NPU 기반의 스위치가 특정 하드웨어 상에서만 프로그래밍 가능하도록 개발되었던 방향과 달리, 특정 하드웨어에 종속되지 않고 독립적으로 프로그래밍할 수 있는 것을 목적으로 한다. 이를 통해 네트워크 장비의 세부적인 구조 모델에 대한 이해가 없더라도 사용자의 요구사항에 맞게 스위치 등을 프로그래밍하는 것이 가능하다. 또한 P4는 네트워크 장치의 하위 레벨 인터페이스에 대한 상위 수준의 추상화 기능을 제공하며 특정 프로토콜에 한정되지 않는 특성을 지닌다.

P4는 그림 1과 같이 네트워크 패킷 처리를 위한 과정을 패킷 파싱 (Parsing)과 테이블 매칭/액션의 제어 플로우의 형태로 표현한다. 즉, 네트워크 장치의 입력 포트로 패킷이 도착하면 해당 네트워크 장치는 개발자에 의해 기정의된 패킷 헤더 파서를 통해 패킷의 헤더 필드를 추출하고 이를 바탕으로 매칭/액션 테이블에 정의되어 있는 액션에 따라 패킷을 처리하게 된다. 또한 P4는 이미 운용중인 장비에 대해서도 새롭게 프로그래밍할 수 있는 API를 제공하는 재구성 기능을 갖고 있고, 오픈플로우 또는 규칙 전달 인터페이스를 통해 새로운 매칭 규칙을 전달할 수 있는 API를 가지고 있다.

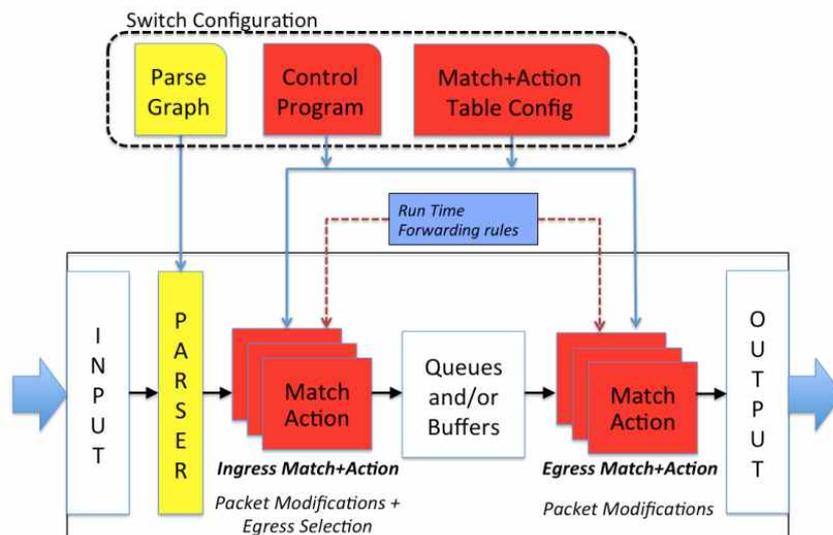


그림 1. P4의 추상화된 패킷 전달 모델

P4를 이용한 프로그램 설계 과정은 다음과 같다. 우선 설계한 프로그램에서 다루게 될 패킷의 헤더 필드를 정의한다. 그리고 정의한 헤더 필드들을 어떤 순서와 규칙에 따라서 패킷에서 추출할지를 패킷 파서를 통해 결정한다. 그런 다음 추출한 헤더 필드의 각 부분에 대해서 어떤 규칙을 적용할지는 매칭/액션 테이블 기반으로 정의하게 된다. P4를 통해 설계된 프로그램은 front-end와 back-end의 2단계 컴파일 과정을 거친다. Front-end 컴파일 과정에선 P4로 작성된 프로그램이 P4 규격에서 정의한 규칙에 맞게 설계되었는지를 검증하는 단계이다. Front-end 컴파일과정은 P4에서 지원하는 P4-HLIR (High Level Intermediate Representation) 프로젝트를 통해 진행할 수 있으며, front-end에 대한 결과물로 P4로 설계된 헤더, 파서, 테이블에 대한 정보가 파이썬 언어 기반의 사전 (dictionary) 형태로 제공된다. 반면, Back-end 컴파일은 대상이 되는 하드웨어에 대해서 front-end 컴파일의 결과물을 매핑하는 과정을 의미한다. P4-HLIR 프로젝트에서는 back-end 컴파일을 위한 테이블 간의 의존성 그래프 (dependency graph)에 대한 정보는 제공해 주지만, 세부적인 back-end 컴파일과정은 각 스테이지 별 파이프라인 구성, 메모리 할당 등과 같은 대상 하드웨어의 특성에 따라 달라지기 때문에 이에 대한 사전 지식을 필요로 한다.

앞서 설명한 것과 같이 P4 언어 자체는 front-end 결과물만 제공하기 때문에 P4는 대상이 되는 장치에 독립적이라고 할 수 있다. 하지만, P4를 이용하여 상용화 수준의 네트워크 스위치를 개발하는 과정에서는 필연적으로 대상이 되는 장치의 규격, 처리 능력을 고려해야 하기 때문에 대상 장치에 대한 의존성을 완전히 배제할 수는 없다. 또한 P4에서는 상태 정보 동기화가 정의한 테이블 수준으로 제한되기 때문에 전역 수준의 상태 정보 동기화가 필요한 경우에는 사용되기 어렵다. 그리고 기본적인 스위칭 기능 이외의 기능들 (예를 들어, 방화벽과 같은 미들박스 기능들)을 제공하기에는 P4 기능이 제한적이기 때문에 P4 컨소시엄에서 지속적으로 언어의 규격 확장에 대한 연구가 진행되고 있다.

정리하면, 특정 네트워크 장비 회사가 일방적으로 제공하는 장비에 의존적이지 않고 네트워크 관리자의 요구사항에 맞게 유연하게 데이터 평면을 프로그래밍할 수 있는 기술이 주목 받고 있으며 대표적인 데이터 평면 프로그래밍 언어인 P4가 공개되어 활발히 연구 개발이 진행되고 있는 상태이다. 하지만, 연구 초기 단계로 제공하는 기능이 제한적이며 개선할 이슈가 다수 존재하기 때문에 향후 프로그래밍 가능한 데이터 평면 기술에 대한 관심과 연구개발이 더 필요할 것으로 사료된다.

[참고문헌]

[1] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming Protocol-Independent Packet Processors," ACM SIGCOMM Computer Communication Review, July 2014.