Contribution ID: **146** Type: **not specified**

P4C-XDP: Programming the Linux Kernel Forwarding Plane Using P4

Tuesday, November 13, 2018 2:00 PM (35 minutes)

The eXpress Data Path (XDP) is a new kernel-feature, intended to provide fast packet processing as close as possible to device hardware. XDP builds on top of the extended Berkely Packet Filter (eBPF) and allows users to write a C-like packet processing program, which can be attached to the device driver's receiving queue. When the device observes an incoming packet, the user-defined XDP program is triggered to execute on the packet payload, making the decision as early as possible before handing the packet down the processing pipeline.

P4 is a domain-specific language describing how packets are processed by the data plane of a programmable network elements, including network interface cards, appliances, and virtual switches. It provides an abstraction that allows programmers to express existing and future protocol format without coupling it to any data plane specific knowledge. The language is explicitly designed to be protocol-agnostic. A P4 programmer can write their own protocols and load the P4 program into P4-capable network elements.
As high-level networking language, P4 supports a diverse set of compiler backends and also possesses the capability to express eBPF and XDP programs.

We present P4C-XDP, a new backend for the P4 compiler. P4C-XDP leverages XDP to aim for a high performance software data plane. The backend generates a eBPF-compliant C representation from a given P4 program which is passed to clang and llvm to produce the bytecode. Using conventional eBPF kernel hooks the program can then be loaded into the eBPF virtual machine in the device driver. The kernel verifier guarantees the safety of the generated code. Any packets received/transmitted from/to this device driver now trigger the execution of the loaded P4 program.

The P4C-XDP project is an open source project hosted at https://github.com/vmware/p4c-xdp/. We provide prove-of-concept sample code under the tests directory, which contains a couple of examples such as basic protocol parsing, checksum recalculation, multiple tables lookups, and tunnel protocol en-/decapsulation.

Presenters: RUFFY, Fabian (University of British Columbia); BUDIU, Mihai (VMware); TU, William (VMware)

Session Classification: Networking Track