Timestamp-Aware RTP Video Switching Using Programmable Data Plane

Thomas G. Edwards
FOX Networks Engineering & Operations
thomas.edwards@fox.com

Nick Ciarleglio
Arista Networks
nick@arista.com

Abstract

Programmable data plane pipelines are now available in line rate packet processors implemented in commercially-available Ethernet switches. The programmable pipeline can be accessed to provide a range of novel functionalities, including new applications for professional media IP networks. This demo shows the use of programmable data plane to obtain "clean" video switching of (pre-standard) SMPTE ST 2110-20 uncompressed HD video flows based on the RTP timestamp. This application was originally developed in the P4 programming language, and ported for implementation to the Cavium XPliant network processor on the Arista 7160 Ethernet switch.

Description

The broadcast industry is now preparing for the carriage of live professional media over IP. The primary goal of this effort is to enhance the flexibility and agility of the video production plant. Particular advantages include: agnosticism to resolution, bit depth, and frame rate, compatibility with network interfaces on commodity Ethernet switches and commodity servers, flexible assembly and association of media streams, and a natural mechanism to coordinate with network-based registration and discovery of devices, streams, and media processing capabilities. This change is also expected to provide greater density of switching and virtualization of video processing.

The network requirements of low-latency, uncompressed high definition networked video intended for live production is quite different than the HTTP-delivered adaptive bit rate (ABR) video intended for end-user distribution. Data rates are high, around 1 Gbps. Latency requirements are small, with expectations of 1ms or less end-to-end latency from sender to receiver. Due to these dual requirements, transmissions will tend to use UDP with expectations of reserved bandwidth and zero packet loss.

Software Defined Networks (SDN) gives broadcasters fine-grained programmatic control over their networks to achieve these goals. However, non-programmable data plane SDN APIs (such as OpenFlow) are limited by implementation decisions made in the hardware data plane, such as a fixed number and format of packet headers, and a fixed set of actions that can result from match/action tables. Also communication between the data plane and control plane results in a high level of latency between control decisions and re-configuration of the data plane.

Programmable data plane technology enables a more flexible method of controlling packet processing that can be performed at line-rate. The packet processor parser can be programmed to extract any header desired, and match/action tables based on those headers can be more complex than in non-programmable switching solutions.

In this demo, an example of line-rate video switching is shown. When video sources are switched in professional productions, it is important that the switch be done in a “clean” fashion at the boundary between frames to avoid undesirable artifacts.

Draft standard SMPTE ST 2110-20 is an RTP format for the carriage of uncompressed video images. It has the quality that every RTP packet in a frame carries the same RTP timestamp (although all packets have unique RTP sequence numbers). This means that a programmable data plane pipeline that makes packet switching decisions based on the RTP timestamp can also be used as a clean video switching system.

Two synchronized sources of high definition video with a 1280x720 pixel progressive image of 59.94 frames per second provide video over a serial digital interface (SDI). These inputs are converted into SMPTE ST 2110-20 uncompressed video RTP streams over 10 GbE using a Nevion VS902 converter system. The Ethernet streams are then sent into an Arista 7160-32CQ Ethernet switch, which utilizes the programmable data plane Cavium XPliant network processor to perform the RTP timestamp-aware video flow changes.

Code for this application was originally developed in the P4 programming language, where it could be tested in software using the P4 behavioral model. The P4 code can be found at https://github.com/FOXNEOAdvancedTechnology/ts_switching_P4. Once the basic concept of operation was proven in software simulation, the application was ported for implementation to the Cavium XPliant.