Realizing Source Routed Multicast Using Mellanox's Programmable Hardware Switches

Matty Kadosh[1], Yonatan Piasetzky[1], Barak Gafni[1],
Lalith Suresh[2], Muhammad Shahbaz[3], and Sujata Banerjee[2]
[1]*Mellanox,* [2]*VMware, and* [3]*Stanford University*

ABSTRACT

We present the first hardware implementation of Elmo [7], a recently proposed source-routed multicast technology that leverages programmable switches. Elmo addresses the long-standing control- and data-plane scalability issues in IP multicast. These scalability bottlenecks have made it impossible to adopt IP multicast at the scale of today's public clouds, despite the demand and pervasive use of point-to-multipoint communication patterns in modern cloud applications (e.g., streaming telemetry, replicated state machines, and distributed machine learning [7]).

Elmo overcomes the control- and data-plane scalability limitations of IP multicast by using source-routing and exploiting the unique characteristics of data-center networks (like symmetric topologies and virtual switches). In Elmo, virtual switches, like PISCES [6] or OVS [5], encode multicast trees inside packets using an efficient and compressed header encoding—for a given data-center network—that can be interpreted at line rate by programmable hardware switches. By using source-routing, Elmo significantly reduces the usage of limited-size hardware group tables, the primary data-plane scalability bottleneck. Elmo absorbs the processing and update overhead of physical switches by configuring multicast groups at the virtual switches, which overcomes the control-plane scalability bottleneck.

Central to Elmo's approach is the encoding of the output port list at every hop of a multicast tree using a bitmap, in a data-center network. This design aligns with how existing switching ASICs handle multicast; for example, using group tables, wherein a group ID (e.g., destination IP address) is converted into an output port bitmap, which indicates the set of egress ports for a packet. Elmo's feasibility hinges on hardware switches being able to obtain this bitmap directly from a packet header, and bypassing the pipeline table lookup. While feasible, this idea of an output-port-bitmap primitive was not demonstrated using hardware switches in the Elmo paper [7], because existing switching ASICs [2] do not have a programmable multicast pipeline.

In this work, we leverage Mellanox's Spectrum-2 hybrid programmable switching ASIC [1] to support Elmo. The hybrid switch retains its legacy forwarding pipeline (bridging and routing), while allowing specific programmable logic to be inserted at certain locations (Figure 1).

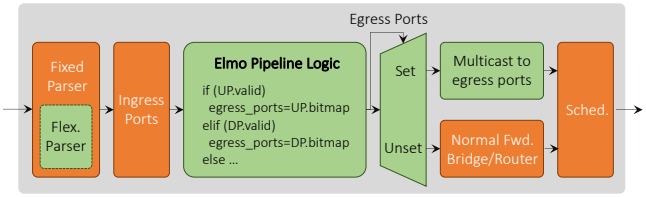

Figure 1: Mellanox's Spectrum-2 hybrid switching pipeline for Elmo. (Orange blocks are fixed hardware logic, while green blocks are programmable.)

To implement Elmo, we solved several challenges. We needed a flexible header format that could be expressed in P4 [3] and can be efficiently parsed by the switch ASIC pipeline. Second, we leverage the capability of Spectrum-2 to implement the output-port-bitmap primitive, which sets egress port list from the parsed bitmap.

Our efforts have also led us to identify several areas where we could improve the P4 language. First, multicast is not a first-class citizen in P4. Second, in the core library, extraction is defined on a header and not on individual fields, which can be costly for hardware in terms of bits needed to be carried along the pipeline. Moreover, defining extraction on fields can allow for more advanced features, like variable offset fields used in Elmo and other protocols (e.g., SRv6 [4]). Lastly, we explored existing solutions for options parsing, needed to parse Elmo and other well-known headers (like IP and TCP options, and more). We found that P4 language specification lacks a clean and simple way to compile options parsing to hardware switches (like Mellanox's Spectrum-2 switch and others).

To evaluate our implementation, we built a testbed with four switches, connected in a Clos-like topology; the topology consists of three leaf switches and one spine switch. Each leaf switch connects to two hosts, with six hosts in total. With Mellanox's Spectrum-2 switch, we can process Elmo at line rate, and emit packets according to the bitmaps specified by the Elmo headers (both unicast and multicast).

In summary, in our talk, we will (1) present a brief overview of Elmo, (2) describe how we implement the protocol using Mellanox's Spectrum-2 switching ASIC, (3) present an end-to-end demo, and (4) describe various P4 language challenges we faced in this effort, and the solutions we propose.

REFERENCES

[1] Mellanox's Spectrum-2 Ethernet Switching ASIC. https://www.mellanox.com/products/ethernet-switch-ic/spectrum-2.

[2] Tofno 2: Second-Generation of World's Fastest P4-Programmable Ethernet Switch ASICs. https://barefootnetworks.com/products/brief-tofino-2/.

[3] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., et al. P4: Programming Protocol-Independent Packet Processors. *ACM SIGCOMM Computer Communication Review (CCR) 44*, 3 (2014), 87–95.

[4] Filsfils, C., Garvia, P., Leddy, J., Voyer, D., Matsushima, S., and Li, Z. SRv6 Network Programming. *Internet-Draft* (2017).

[5] Pfaff, B., Pettit, J., Koponen, T., Jackson, E. J., Zhou, A., Rajahalme, J., Gross, J., Wang, A., Stringer, J., Shelar, P., and et al. The Design and Implementation of Open VSwitch. In *USENIX NSDI* (2015).

[6] Shahbaz, M., Choi, S., Pfaff, B., Kim, C., Feamster, N., McKeown, N., and Rexford, J. PISCES: A Programmable, Protocol-Independent Software Switch. In *ACM SIGCOMM* (2016).

[7] Shahbaz, M., Suresh, L., Rexford, J., Feamster, N., Rottenstreich, O., and Hira, M. Elmo: Source Routed Multicast for Public Clouds. In *ACM SIGCOMM* (2019).