Cooperation Between Control Planes

Srinivasan Seshan (srini@cs.cmu.edu)

Applications such Internet video and cloud computing have becoming increasingly demanding users of the network. As the network demands and user performance expectations continue to grow, we expect that careful control-plane based optimization of these applications will become increasingly critical. We recently studied the problem of making traffic engineering practical for CDNs. As part of this study, we designed a new system, called video delivery network (VDN), that uses two control planes - a centralized plane that performs end-to-end optimization of live stream routing and a distributed control plane that ensures high availability and responsiveness. Running two control loops in tandem can lead to challenges that destroy any benefit that either control loops would have had individually. One of the key contributions of the VDN framework is enabling these two control planes to co-exist in a hybrid control plane approach that balances the benefits of centralized control and distributed control. This was done by: 1) providing a strict prioritization mechanism for choosing decisions from either the centralized or distributed control plane, 2) ensuring that the combined route selection was always loop-free, and 3) providing a simple interface by which each control plane could ensure that its resource allocation decisions did not impact the other control plane.

While VDN solved several important challenges in scaling, availability and performance, we realized that the end result was only a piece of the end-to-end video delivery optimization challenge and that the delivery infrastructure for today's popular video content was surprisingly sophisticated. Typically, video delivery involves the content provider (e.g. HBO), the CDNs that store the video (e.g. Akamai), a video player on the client machine, delivery optimization services (e.g., Conviva) and the ISPs that provide connectivity between these participants. Each of these parties independently tries to optimize the quality of service it provides to users. For example, the video optimization service informs client players which CDN to use based on the service's estimate of which CDN will have the best performance. The ISPs in the Internet make traffic engineering decision based on observed load, i.e., on traffic forwarded to them by other ISPs or CDNs. Finally, each CDN assigns each client request for a video chunk to a CDN cluster based on many factors (geographic proximity, server load, estimated network throughput, etc.) and also selects what ISP to use based on its estimated throughput to the client. Finally, the player software selects the bit-rate variant of the content that it feels is best suited to its estimate of the bandwidth to the chosen CDN node. Since there is no explicit coordination, the end result is a complex system with performance that is often far from optimal. Even stability can be an issue since changes made by one entity among the collection of can change the observations of other parties, and, as a result, a sequence of updates.

As we can see from the above, the control planes that sophisticated applications, such as video delivery and cloud computing, use to manage computation, storage and communication and those that they use to manage network connectivity currently operate in a completely independent fashion. However, the decisions to perform processing tasks on different machines, store content on different nodes or allocate bandwidth to different flows all have significant impact on each other. As we move forward, we expect the desire for better performance and reliability to drive tighter integration between these control planes – both in the form of coordination between them and control planes that manage multiple of these resources in a more integrated fashion.

The problem of coordinating multiple control planes is that we lack the appropriate interfaces between control planes. Ideally, increasing the visibility of each control plane into the decision process of its peers would help greatly. However, in practice this is difficult for a number of reasons. First, competitive concerns limit the amount of information any participant is willing to release. For example, the BGP protocol provides only very simple protocol features, such as MED (Multi-Exit Discriminator), for routing domains to communicate preferences to each other. Second, it is likely unreasonable for every participant to understand the metrics and tradeoffs of its peers. This is especially true in designing interfaces between control planes for different types of resources. For example, it has proven difficult to define standard metrics between routing domains and it is likely to be even more difficult to define metrics that CDNs and ISPs can agree to. This does not mean that defining cross-domain control plane interfaces are hopeless. There is an end-to-end application that ties these applications together and recent efforts in understanding user QoE provide a target optimization goal. Similarly, the parties involved in an application often have pairwise arrangements that allow them to exchange information about costs and service guarantees. Making small changes to these interfaces focused on end-to-end performance may provide significant benefits. In addition, exploring the design of these interfaces in a few well-defined settings such as video delivery and cloud computing can help identify the key properties needed in any cross-domain setting.

Brief Bio: Srini Seshan has a long history of working on various parts of the Software Defined Infrastructure from the lowest layers of the protocol stack to the very top layers. This includes work on; 1) defining programming interfaces and functionality partitioning for software defined radios [NSDI 2009], 2) designing control systems for selecting routes in multihoming settings [Sigcomm 2004]; and 3) hybrid centralized/distributed control systems for managing video delivery to clients across a wide-area network [Sigcomm 2015].