Scaling Up Clustered Network Applications with ScaleBricks – Public Review

Bruce Davie
VMware, Inc.
Palo Alto, CA, USA
bdavie@vmware.com

The idea of using clusters of commodity processors to perform packet processing is at this point a well-established technique. RouteBricks is probably the most well-known example of this approach. The telecommunications industry is now embracing the idea that many tasks that have traditionally been performed on specialized hardware might reasonably be performed on commodity processors.

Because a single commodity processor typically forwards packets at a much lower rate than an ASIC, it's often necessary to deploy clusters of processors to perform tasks that had previously been performed by a single hardware appliance. This raises a number of challenges around scalability—ideally, adding more processor nodes should increase the capacity of the system linearly.

ScaleBricks is motivated by the observation that "capacity" is a multi-faceted quantity. Not only does the cluster need to increase in throughput as nodes are added but, in some applications, the number of forwarding entries that it can handle also needs to scale up. The authors refer to this as FIB (Forwarding Information Base) scaling. The update rate of the FIB should also scale up. This leads to a design where the FIB is partitioned across the nodes—no single node holds the entire FIB. With such a design, each packet will potentially hit a node in the cluster that doesn't have all the forwarding information for that packet. This leads to the central design challenge of the paper: to design efficient algorithms to forward a packet from the ingress node to the node that has the full information necessary to process the packet (the handling node).

In addition to scaling, forwarding latency is also a design consideration. The partitioning of the FIB requires that most packets are touched by at least two nodes (ingress node and handling node). The ScaleBricks design seeks to bound the number of nodes that touch the packet to two, using a hardware switch to forward packets from ingress node to handling node. This contrasts with alternative approaches that use intermediate processor nodes to forward packets across the cluster.

Lest this design space seem arbitrarily constrained, the paper uses a motivating example from a real product, the cellular network-to-Internet gateway of an LTE mobile network. Such gateways maintain per-flow state at the handling nodes, and also are constrained in the allocation of flows to handling nodes by external factors. Thus the problem of getting an incoming packet to its correct handling node is very real in this scenario, and can't be easily worked around (e.g., by using a simple hash to assign flows to nodes).

With this setup, the paper then works through a clever series of innovations to solve the stated problem: figure out how to map an incoming packet to its correct handling node without storing a complete FIB at every node. The key to this approach is to use a "global partition table" (GPT)—a compact data structure that can efficiently direct each incoming packet to the correct handling node, using much less space than a full FIB would require. The details of how this table is built and why it works are clearly spelled out in the paper.

There remain a few limitations. First, the FIB partitioning approach does not scale indefinitely. At small numbers of nodes, FIB capacity scales almost linearly with node count, but drops off quickly, with no improvement at all after 32 nodes. This suggests the approach will be valuable for some applications but not others, depending on the FIB requirements.

It's also worth noting that update rates remain a possible area of concern. While partitioning the FIB across nodes should improve update rates relative to FIB replication (only one node needs to be updated for a given FIB change), the GPT structure is not especially efficient at handling updates.

The largest concern for the PC was the breadth of applicability. While the work is presented as being quite general, it does seem that the constraints on the design space must line up in a particular way for ScaleBricks to be the right answer. The authors rightly note that finding other applications that will benefit from this interesting system design is an area for further investigation. ScaleBricks looks to be an innovative packet processing system in search of further applications.