Architecture and Algorithms for Virtual Routers as a Service

Zdravko Bozakov
Institute of Communications Technology
Leibniz Universität Hannover
Email: zdravko.bozakov@ikt.uni-hannover.de

***Abstract*—The deployment of virtualized network resources has the potential to spur new business models and increase flexibility for network customers as well as infrastructure operators. It is worthwhile to re-evaluate how to effectively express traditional network elements in the virtualization domain. In this paper we consider network routers and argue that the representation of routing functionality as a service, rather than an isolated virtual resource is better suited in the virtualization context.**

We present an architecture enabling physical infrastructure operators to provide routing as a service by combining distributed forwarding elements to appear a single virtual router instance which routes traffic between a set of customer points of presence. We provide embedding algorithms for virtual router topologies with minimum allocation cost. We consider the customer's geographical attachment to the network, bandwidth demands as well as capacity constraints in the core substrate.

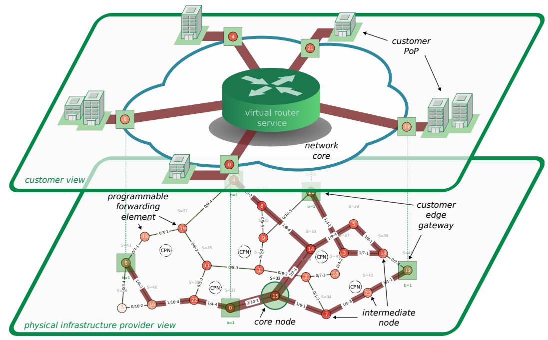
Parts of this work have been published at the International Workshop on Quality of Service 2011 (IWQoS'11).

I. Introduction

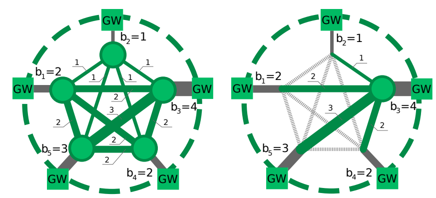
To date, a substantial amount of research in the network virtualization domain has focused on the embedding of predefined virtual network topologies onto a physical substrate, a problem known to be NP-hard. A number of heuristics for the general substrate embedding problem approximating the optimal solution have been proposed e.g. [1], [2], [3].

In this paper we advocate the concept of virtual routers as a service - a collection of virtual network resources functioning as a single router instance as illustrated in Fig.1a. We believe that routing functionality in virtual networks is more suitably defined in terms of connectivity between end points rather than topologies mimicking physical networks. Traditional design goals such as resilience are likely to remain a responsibility of the physical infrastructure provider, addressed independently of the virtual domain instantiation. An inherent advantage of this simplified viewpoint is that the substrate embedding problem becomes tractable. We discuss algorithms for the *optimal* allocation of resources in capacity constrained substrate networks. In addition, we develop a flexible architecture for virtual router services (VRS).

VRS can be deployed to consolidate physical provider resources and adapt substrate allocation to changing network conditions without disrupting running services. At the same time, customers can reduce the number of physically hosted devices while seamlessly integrating their router instance into an existing infrastructure. Additional aspects of the single router abstraction as a means for facilitating network management are discussed in the position paper [4].



(a) A virtual router service connecting five customer locations with a specific capacity demand over a provider substrate.



(b) Fully connected topology (c) Star topology

Fig. 1. Equivalent VRS embedding: allocation cost S=17 (b) and S=8 (c).

II. Embedding Virtual Router Services

Our architecture is based on the assumption that customers expect the functionality of a virtual router service to be indistinguishable from that of a physical device, i.e. the traffic flow between any two nodes attached to the router is limited only by the capacity of their interfaces and routing tables are calculated by a single routing process. Geographical attachment of customer PoPs, corresponding capacity demands as well as the available bandwidth in the substrate are the primary constraints for VRS.

In the following we consider bandwidth allocation costs for a VRS connecting a set of customer PoPs N with capacity demands b_u for $u \in N$. We define the VRS allocation cost S as the sum of reserved substrate bandwidths b, weighted by the respective link costs c. Without loss of generality, we analyze a fully connected substrate topology spanned between $n = |N|$ edges. In terms of capacity the VRS instances depicted in

Figures 1b and 1c offer equivalent connectivity. In Fig.1b, $\min(b_u, b_v)$ units of bandwidth are reserved between each pair of nodes $(u, v) \in N$. Hence, the allocation cost S_{full} is given by $\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \min(b_i, b_j)c_{ij}$. Setting the capacity demands and link costs to one, it becomes evident that the cost increase is quadratic: $S_{full}^1 = n(n-1)/2$. Hence the use of a point to point VRS allocation scheme is problematic even for relatively small numbers of PoPs.

On the other hand, if we select any node $k \subset N$ and route traffic from all remaining edges over it, as depicted in Fig.1c, the allocation cost becomes $S_{star} = \sum_{i=1}^{n-1} \min(b_i, b_k)c_{ik}$ and grows linearly with the number of customer edge nodes and $S_{star} < S_{mesh}$ for all $k \in N$. In fact, for an appropriately chosen core node k, a star topology provides the overall least cost connectivity between a set of edge nodes in an arbitrary substrate network.

III. Virtual Router Service Architecture

Based on the cost considerations above we propose a star architecture comprised of a single core node, responsible for all Layer 3 routing decisions, connected to a set of customer edge gateways (CEG) over a series of intermediate nodes (IN). Each VRS is associated with a unique control plane instance running in a virtual machine (VM) hosted at a suitable network site. Each VM controls its associated network elements over a dedicated link. The architecture relies on a programmable network substrate which allows a VRS controller to modify the L2 and L3 flow tables of all associated forwarding engines (e.g. [5]). We extend the architecture proposed in [6] to include the setup and management of paths connecting the VR core to customer PoPs at the network edge. The selection of least cost paths and the installation of the corresponding forwarding entries is performed by a path management controller (PMC).

IV. Algorithms for Virtual Router Service Embedding

Embedding a VRS involves two independent operations: the selection of an optimal core node location and the allocation of optimal forwarding paths to the CEGs.

Path Selection: The goal of the operation is the identification of least cost paths connecting the core node r to a set of CEGs E while providing sufficient capacity. In substrate network with limited capacity, a basic shortest path approach is not guaranteed to minimize the allocation cost.

We formulate the VRS path allocation task as a flow network problem, which can be solved using a minimum cost flow (MCF) algorithm. We interpret the substrate graph G as a flow network and define the CEGs as traffic sinks with a flow demand of b_e and the core node r as a traffic source with a flow supply of $b_r = -\sum_E b_e$. An *optimal* set of paths w.r.t. to any given core r can be calculated using the successive shortest paths (SSP) [7] algorithm, among others. The SSP algorithm has the advantage that it can efficiently handle edge demand changes or attachment of new CEGs. Note that the optimal flow may be split along multiple paths as proposed in [2] if demands $b_i \neq b_j$ for $(i, j) \in E$.

Algorithm 1 VRS embedding

prune nodes with insufficient resources
$S_\infty \leftarrow \infty$ // initialize array of lower bound costs
for $e \in E$ **do** // iterate through all edge nodes
  get shortest path distances $d(n)$ from e to all $n \in G_\infty$
  $S_\infty(n) \leftarrow S_\infty(n) + d(n)b(e)$
end for
sort S_∞ by ascending cost
$s_{min} \leftarrow \infty$, $r_{min} \leftarrow \emptyset$
$(n, s) \leftarrow \text{pop}_0(S_\infty)$ // remove least cost node/cost tuple
while $s > s_{min}$ **do**
  $s_{min} \leftarrow$ SSP(n,G), $r_{min} \leftarrow$ n
  $(n, s) \leftarrow \text{pop}_0(S_\infty)$
end while
return r_{min}, s_{min}

Core Node Selection: The choice of the core node location is vital to ensure a minimum cost VRS allocation. To avoid checking every feasible core node candidate for optimality using the SSP algorithm, we consider the uncapacitated instance of the substrate graph G_∞. We then calculate the allocation costs $S_\infty(n)$ for all $n \in G_\infty$ using Dijkstra's algorithm and use these as a lower bound for the capacity constrained case as outlined in Alg. 1. Our simulations confirm that this approach substantially reduces the number of required iterations.

V. Conclusion

We outlined an architecture for virtual router services which transparently manipulates the forwarding tables of a set of distributed devices allowing them to be operated as a single entity. The VRS takes advantage of the programmability offered by state-of-the-art network components. By defining the VRS in terms of customer edge capacity demands, the calculation of optimal substrate mappings is made possible. We presented algorithms for a minimum cost VRS embedding in capacity constrained substrate networks. The ability to efficiently allocate VRS instances and migrate resources on the fly paves the way for attractive new business models while ensuring a simplified deployment and operation.

References

[1] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," in *Proc. IEEE INFOCOM 2006*, Apr. 2006, pp. 1 –12.

[2] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *SIGCOMM CCR*, vol. 38, pp. 17–29, March 2008.

[3] N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proc. IEEE INFOCOM 2009*, Apr. 2009, pp. 783 –791.

[4] E. Keller and J. Rexford, "The "platform as a service" model for networking," in *Proc. INM/WREN '10*, April 2010.

[5] N. McKeown et. al., "Openflow: enabling innovation in campus networks," *SIGCOMM CCR*, vol. 38, no. 2, pp. 69–74, 2008.

[6] Z. Bozakov, "An open router virtualization framework using a programmable forwarding plane," in *Proc. ACM SIGCOMM*, 2010, pp. 439–440.

[7] R. K. Ahuja, T. L. Magnati, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993, pp. 320–324.