Service Characterization for Virtual Routers

Zdravko Bozakov
Leibniz Universität Hannover, Germany
zdravko.bozakov@ikt.uni-hannover.de

I. Introduction

In the past decade the virtualization paradigm has been remarkably successful in the server domain, and significant benefits are expected from the virtualization of network resources. However, to date the virtualization capabilities of proprietary systems remain limited while the impact of executing open software routers within system virtualization platforms has not been fully investigated.

In this talk we present approaches for router performance evaluation and measurement results comparing the capabilities of various hardware and software routing platforms. Our findings show, that the use of virtualized software forwarding planes is problematic.

Aiming to eliminate this bottleneck, we outline a framework for router virtualization which employs open source software components for the control plane and hardware supporting the emerging OpenFlow protocol as a forwarding plane.

II. Router Service Characterization

Packet throughput is the most commonly used metric for evaluating router performance. However, this metric does not capture characteristics, such as packet delays, jitter, or queueing effects. Network calculus server models, such as Guaranteed Rate (GR) [1] can be highly beneficial for network analysis and the computation of services guarantees. In the following we present methods and metrics suitable for quantifying router platform performance using external measurements. Furthermore, we examine what information about the system architecture can be extracted from the measurements.

Guaranteed Rate Server Model Parameter Estimation: A server belongs to the GR scheduler class [1], if it guarantees that the packet departure time d_i is at most

$$d_i \leq VFT_i + e_i \tag{1}$$

where VFT_i denotes the GR virtual finishing time and e_i is an error term indicating the maximum deviation of the departure time of a single packet compared to an ideal constant rate server. The error term e_i can be estimated directly by investigating the arrival and departure times of each packet traversing the system using Eq. 1. The VFT for a packet with size l_i, a guaranteed service rate r, and arrival time a_i is calculated recursively using

$$VFT_i = \max\{a_i, VFT_{i-1}\} + \frac{l_i}{r}. \tag{2}$$

The relationship between the GR virtual finishing time, error term, and packet arrivals and departures is depicted in Fig. 1. The error term estimate gives an empirical bound for the delay experienced by a packet traversing the router.

Rate Estimation: The service rate of a given flow can be estimated by analyzing packet departure times within a backlog period [2], i.e. for n packets in a given backlog period the rate estimate r is given by

$$r = \frac{\sum_{i=1}^{n} l_i}{d_n - d_0} \tag{3}$$

where l denotes the packet size, d_0 is the departure time of the first, and d_n the departure time of the last packet in the backlog period. The maximum r_{max} of all backlog periods is then taken as the estimate for the allocated rate of a flow [3]. The rate estimated using the smallest possible packet size l_{min}, is related to the maximum packet per second rate r_{pps} which a router with nominal capacity C can process:

$$r = \min\left(r_{pps} l_{min}, C\right) \tag{4}$$

Error Term Estimation: The maximum processing delay δ_{max} of all backlog periods with only one packet in backlog is taken as an estimate of the error term e. Assuming negligible propagation delays and empty system queues the packet processing time is calculated using

$$\delta_i = d_i - a_i - \frac{l_i}{C}. \tag{5}$$

As shown in the results section, the processing delay can show a strong dependency on the packet size.

III. Results

We used the setup depicted in Fig. 2 to analyze several routing platforms using the methods described above. We measured the performance of a Cisco 7204 VXR router, a NetFPGA router [4], a Linux software router, a virtualized software router using Xen [5], and VServer/PlanetLab slices. Data was captured using an Endace DAG card which provides nanosecond packet timestamping precision. Fig. 3(a) depicts

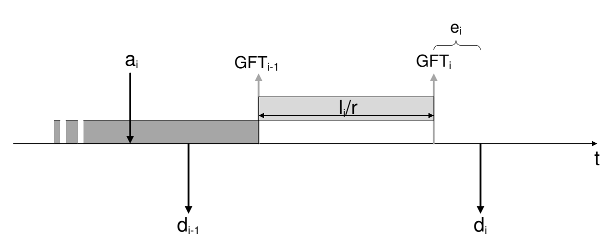


Fig. 1. GR server Virtual Finishing Time (VFT) and error term.

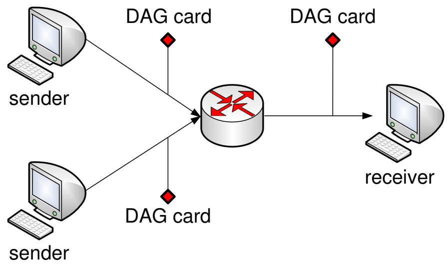
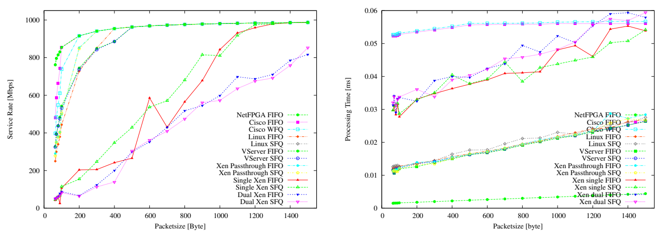


Fig. 2. Experimental Setup



(a) Packet throughput

(b) Packet processing time

Fig. 3. Measurement results for different router systems.

the maximum achievable rates of the tested systems for UDP CBR traffic, and different packet sizes. Fig. 3(b) shows the processing times of all tested systems. Additional, in-depth results can be found in [6].

It is noteworthy, that compared to the Cisco hardware, the software routers exhibit lower packet processing times, although the forwarding performance of the latter is worse. On the other hand, the delay variance of the Cisco router was significantly lower. The poor Linux forwarding performance indicates that the Kernel packet processing operations introduce a constant delay between the transmission of successive packets. For the given Linux setup, the delay is approximately $1.6\ us$ which corresponds to a rate of $600K$ packets per second. For Xen this delay increases to $\sim 10\ us$, limiting the forwarding rate for 64 Byte packets to 50 Mbps. The processing delays of the virtualized routers is approximately two times the delay of a pure Linux system. Additionally our measurements showed a linear dependency between the processing delay and packet size, which is due to the packet transmission time over the system bus. Moreover we found that NIC interrupt generation rate is directly related to packet jitter. Generally, our measurements show a significant performance disadvantage for software routers.

IV. Beyond Virtual Software Routers

As the performance of fully virtualized software routers is problematic, we propose a virtual router architecture, depicted in Fig. 4, which employs off-the-shelf Openflow-enabled switches functioning as a programmable forwarding plane and a virtualized control plane running on commodity hardware using an open source hypervisor. Virtual router (VR) instances are represented by a virtual environment (VE) running the control plane, connected to the forwarding plane hardware over

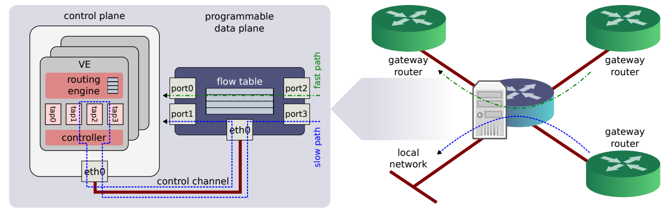


Fig. 4. Virtualization Framework

a control channel. Within each VE a virtual router controller (VRC), transparently maps the virtual interfaces of the router instance to the physical ports of the data plane. Additionally, the VRC mirrors the VE routing tables onto the data plane by installing appropriate Openflow rules. As the data plane communication is hidden from the VE, the VR routing tables can be populated using arbitrary routing daemons. Moreover, the operating system networking stack can be utilized e.g. for processing control and address resolution messages. A major advantage of this architecture is that the virtualization of the forwarding plane is shifted outside the software hypervisor, thereby substantially reducing overhead.

Prototype Implementation: We evaluated the feasibility of the above approach, by implementing a prototype using commodity PC hardware and a NetFPGA card with a reference bitfile implementing version 0.9 of the Openflow protocol. The NOX framework [7] was used for the VRC implementation. Our prototype employs the KVM hypervisor for the control plane virtualization and the XORP suite for routing. Within each VE we used the Linux TUN/TAP framework to instantiate the virtual network interfaces.

V. Conclusion

We presented methods for router performance evaluation as well as measurements describing a range of router characteristics. Our results highlight the difficulties of purely software-based router virtualization. To circumvent these problems, we propose a framework for virtualization which shifts the forwarding plane to Openflow-enabled hardware.

References

[1] P. Goyal, S. S. Lam, and H. M. Vin, "Determining end-to-end delay bounds in heterogeneous networks," *Multimedia Syst.*, vol. 5, no. 3, pp. 157–163, 1997.

[2] A. Kuzmanovic and E. W. Knightly, "Measuring service in multi-class networks," in *In Proc. of IEEE INFOCOM*, vol. 3, 2001, pp. 1281–1289.

[3] A. Undheim, Y. Jiang, and P. Emstad, "Network calculus approach to router modeling with external measurements," in *In Proc. of CHINACOM*, Aug. 2007, pp. 276–280.

[4] J. Naous, G. Gibb, S. Bolouki, and N. McKeown, "Netfpga: reusable router architecture for experimental research," in *In Proc. of ACM PRESTO*. New York, NY, USA: ACM, 2008, pp. 1–7.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *SOSP '03: Proceedings of the 19th ACM symposium on Operating systems principles*. New York, NY, USA: ACM Press, 2003, pp. 164–177.

[6] M. Bredel and Z. Bozakov, "Analyzing router performance using network calculus with external measurements," in *In Proc. of IWQoS*, 2010.

[7] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "Nox: towards an operating system for networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, 2008.