Control and Forwarding Plane Interaction in Distributed Routers

Markus Hidell, Peter Sjödin, and Olof Hagsand

KTH – Royal Institute of Technology
ELECTRUM 229, SE-164 40 Kista, Sweden
{mahidell, psj, olofh}@imit.kth.se

Abstract. The requirements on IP routers continue to increase, both from the control plane and the forwarding plane perspectives. To improve scalability, flexibility, and availability new ways to build future routers need to be investigated. This paper suggests a decentralized, modular system design for routers, based on control elements for functionalities like routing, and forwarding elements for packet processing. Further, we present measurements on the distribution of large routing tables in an experimental platform consisting of one control element and up to 16 forwarding elements.

1 Introduction and Related Work

The growth of the Internet in combination with the demand for new services rapidly increase the requirements imposed on network systems, such as IP routers. The growing traffic volumes require higher performance of the *forwarding plane*, i.e., the router's capacity to process and forward packets. New services often require both more operations to be performed per packet, and that new protocols are introduced in the routers. The latter fact leads to an increased complexity in the *control plane*, i.e., the software controlling the router.

We believe that the monolithic structure of traditional routers is an architectural limitation when it comes to meeting future requirements. Therefore we take the approach to investigate architectures that allow network systems to be *composed* from multiple *modules* (or *elements*), which communicate through open, well-defined interfaces. Our hypothesis is that such architectures can significantly improve scalability, flexibility, and availability of routers. However, there is a certain cost associated with the decentralized structure, since the internal communication incurs overhead. In this paper we investigate different aspects of the internal communication.

A considerable amount of work has been done on modularization and programmability of network systems in the context of active and programmable networks [2], [3], with the purpose to dynamically modify the packet processing path. Exploring decentralized architectures is in line with both industry and research efforts to improve the scaling of Internet routers. Recent commercial high-performance routers are based on distributed multi-chassis solutions, [6], [7], and the 100 Tb/s router project at Stanford University [5] targets a distributed architecture where line card chassis are connected to an optical switch fabric.

Within the IETF, the ForCES (Forwarding and Control Element Separation) working group [4] aims at defining a protocol for communication between control functions in a router and packet forwarding functions [10]. Goutadier [11] has evaluated an early proposal of the ForCES protocol, called Netlink2.

Finally, Feamster et al. suggest an approach to separation between routing and forwarding [9]. They propose that a *Routing Control Platform* (RCP) should be used to separate interdomain routing from the individual routers, which mainly should be concerned with route lookups and forwarding of packets.

2 System Design and Implementation

We consider a distributed router consisting of different functional elements. Using the terminology of ForCES [4], the distributed router consists of Control Elements (CEs) and Forwarding Elements (FEs). CEs implement functions such as routing protocols, while FEs perform for example packet forwarding. CEs and FEs are connected to an internal network, which can be built in many different ways, e.g., using high-speed optics or high performance switches. In principle, the internal network could even be a router-based IP-network!

Internal communication protocols coordinate activities between the different elements. We have taken the approach to use the protocols emerging within the IETF as a starting point, and added extensions for our specific purposes. We call the result *Forz* – a protocol with three main parts: association, configuration, and data transfer.

The association part is related to internal element discovery and system configuration. The purpose of the configuration part is to convey configuration commands and event notifications between CEs and FEs. For example, if the routing protocols in the CE compute a new best route for some destination, then the FEs need to be informed of this so that they can change their forwarding tables. Forz configuration is based on distributing Netlink [8] messages over the internal control network. Netlink is an API of for Unix networking applications. The purpose of the data transfer part is to switch data packets between FEs across the internal network.

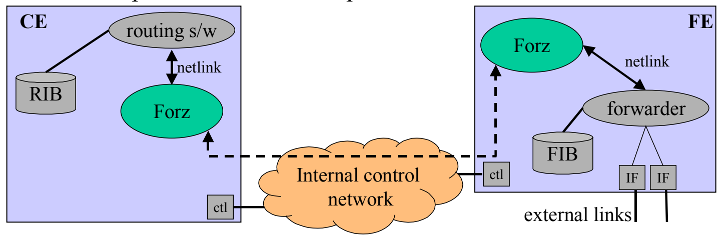

Fig. 1 Physical separation between control element and forwarding element

We have implemented a prototype version of a distributed router consisting of regular PCs connected together by two switched Ethernets (one for control and one for data). So far we have implemented one version of a CE, based on a UNIX system running Zebra open source routing software [1], and one FE implementation, also based on PCs running UNIX. The CE is implemented by extending Zebra with the Forz protocol, and supports the manual configuration of remote network interfaces and static routes. For example, the CE takes commands, given through Zebra's command line interface or a configuration file, and translates them to Netlink messages that are distributed through the Forz protocol (see Fig. 1).

3 Performance Evaluation

The purpose of our performance evaluation is to investigate how different transport mechanisms affect the cost in terms of time spent on internal communication. We study a communication-intensive operation, where a large routing table of 100K entries is distributed from the CE to the FEs. We only measure the communication time between the CE and the FEs. Table updates in the FEs are not included.

Forz can run on top of UDP as well as TCP. The choice of transport protocol depends on the type of information that is communicated over the internal network. The distribution of a routing table means that information is duplicated from a sender (CE) to many receivers (FEs). Thus, it appears to be an ideal candidate for multicast transmission. To investigate this, we compare UDP multicast to two unicast mechanisms: UDP unicast, and TCP (which is always unicast).

In contrast to TCP, UDP lacks mechanisms for flow control, congestion control, and segmentation. So in order to use UDP, we have to add support for those mechanisms in Forz. For example, we quickly discovered that if we use UDP without any flow control, a large portion of the route updates are lost since the receivers (the FEs) cannot process the incoming messages fast enough. Furthermore, we also found that the packet size has significant influence on UDP performance. However, congestion control has not been needed in our measurements, since they have been performed in a controlled environment without congestion.

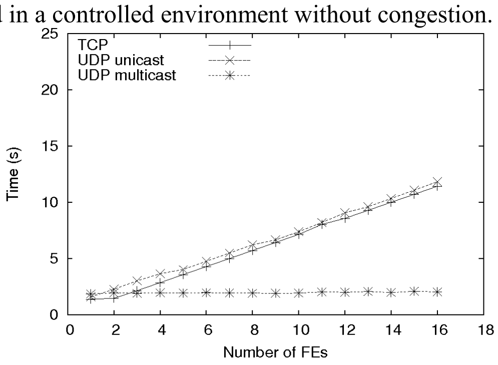

Fig. 2 Total routing table distribution time for different transport mechanisms

We study how the routing table distribution time varies with the number of FEs, and the results are shown in Fig. 2. We observe that, for unicast transport, there is a linear increase in the time it takes to distribute the routing table to all FEs when the number of FEs increases. For UDP multicast, the time is constant, independently of the number of FEs. The value is roughly the same as for UDP unicast with one FE. Thus, in this experimental set-up, the system can distribute roughly 50,000 route updates per second when UDP multicast is used as the transport mechanism.

For comparison, we also measure TCP. Thus, the transport is reliable and there is no need for Forz level ACKs and segmentation. The drawback is that multicast transfers cannot be used. The results show that the total time to distribute the routing table is slightly lower for TCP than for UDP unicast, which can be explained by TCP's more efficient flow control mechanism and more optimal segmentation.

4 Conclusions and Further Work

We have examined how the time for internal communication depends on the number of FEs. From a scaling perspective, the ideal result is an internal communication time that is independent of the number of FEs, as in the case with UDP multicast. Further, mechanisms such as flow control and segmentation are crucial for the performance. For the unicast-based transport mechanisms, the total distribution time increases linearly with the number of FEs. Such an increase is far from ideal, but it is predictable and may in some cases be manageable even for large systems. The implementation of flow control and segmentation is specific to the configuration we have measured. A more general solution would be to use a reliable multicast protocol. Evaluating such protocols for transport of Forz messages is ongoing work.

References

1. GNU Zebra, URL=http://www.zebra.org
2. Computer Networks Special Issue on Programmable Networks, Vol. 38, No. 3, Feb. 2002.
3. IEEE Journal on Selected Areas in Communications on Active and Programmable Networks, Vol. 19, No. 3, Mar. 2001.
4. ForCES (Forwarding and Control Element Separation) IETF Working group, URL=http://www.ietf.org/html.charters/forces-charter.html.
5. I. Keslassy, et al, "Scaling Internet Routers Using Optics", ACM Sigcomm 2003, Karlsruhe, Germany, 2003.
6. Juniper, "T-Series Routing Platforms: System and Packet Forwarding Architecture", White paper, 2002.
7. Cisco, "Next Generation Networks and the Cisco Carrier Routing System", White paper, 2004.
8. J. Salim, A Kleen, and A. Kuznetsov, "Linux Netlink as an IP Services Protocol", Internet RFC 3549, Jul. 2003.
9. N. Feamster, et al, "The Case for Separating Routing from Routers", ACM SIGCOMM FDNA Workshop, Portland, Oregon, USA, Aug. 30, 2004.
10. A. Doria et al, "ForCES Protocol Specification", IETF Internet Draft draft-ietf-forces-protocol-01.txt, Oct. 2004.
11. G. Goutaudier, "Enhancements and Prototype Implementation of the ForCES Netlink2 Protocol", IBM Research Report RZ 3482 (# 99522), Sep. 2003.