VM-based Distributed Active Router Design

Tomáš Rebok

Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno
xrebok@fi.muni.cz

Abstract. The active network approach allows an individual user to inject customized programs into an active nodes in the network, usually called programmable/active routers, and thus process data in the network as it passes through. As the speeds of network links still increase, and subsequently, the applications' demands for the network bandwidth increase as well, a single active router is infeasible to process such high-bandwidth user data in real-time, since the processing may be fairly complex. In order to improve the scalability of such a system with respect to number of active programs simultaneously running on the router and with respect to the bandwidth of each passing stream processed by active sessions, the distribution of processing load and network bandwidth is desirable. The subject of this paper is to propose a distributed active router architecture with a loadable functionality that allows strong isolation of users' programs using Virtual Machine (VM) approach.

1 Introduction

Contemporary computer networks behave as a passive transport medium which delivers—or in case of the best-effort service tries to deliver—data from the sender to the receiver. The whole transmission is done without any modification of the passing user data by the internal network elements[1]. We believe that in the future-generation networks this simple but extremely fast forwarding paradigm will be extended with a layer that will change the network into an active transport medium, which processes passing data based on data owners or data users requests. Multimedia application processing (e.g., video transcoding) and security services (data encryption over untrusted links, secure and reliable multicast, etc.) are a few of possible services which could be provided. The principle called "Active Networks" or "Programmable Networks" is an attempt how to build such intelligent and flexible network using current "dumb and fast" networks serving as a communication underlay. In such a network, users and applications have the possibility of running their own programs inside the network using inner nodes (called *active nodes*, *active routers*, or *programmable routers*—all three with rather identical meaning) as processing elements.

[1] Not including firewalls, proxies, and similar elements, where an intervention is usually limited (they do not process packets' data).

Scalability. The still increasing speed of network links and, subsequently, still increasing applications' demands for higher network bandwidths make a single active router infeasible to process passing user data in real-time, since such processing may be fairly complex (e.g., video transcoding, data encryption, etc.). To be capable of processing higher amount of active programs simultaneously running on the active router and/or processing high volumes of data of each passing stream at high rates, there is the necessity to distribute:

- *processing load*—to enable processing amounts of data that are impossible to process via any single computer,
- *network load*—to avoid bottlenecks formed by networking interface of single processing computer.

The distribution of processing load is efficient only when computationally intensive processing is required. However, the distribution of processing load only is not suitable for many applications because internal architecture of most commonly used IA32 PC architecture with PCI busses is easy to saturate when working with multi-gigabit data flows and thus the network load needs to be distributed over multiple hosts as well.

The goal. The main goal of our work is to propose a VM-based distributed active router architecture with loadable functionality that uses commodity PC clusters interconnected via the low-latency interconnection (so called tightly coupled clusters) to perform distributed processing. In order to achieve reasonable isolation among the users of the active router, the architecture is designed to facilitate implementation based on the virtual machines approach [1].

The rest of this paper is organized as follows: The proposed VM-based distributed active router architecture is described in Section 2 and the architecture of router parallel unit is given in Section 3. The state-of-the-art in the area of distributed active routers' architectures is given in Section 4 and the concluding remarks and proposals for our future work are in Section 5.

2 VM-based Distributed Active Router

In this section we outline distributed AR architecture based on the parallelization of the architecture of single AR described in detail below in Section 3. However, such parallelization introduces problems with sending part of the distributed AR, which may introduce packet reordering. While packet reordering is largely unwanted for general router behavior as it may severely tamper performance of lots of applications and especially those based on the most widely used the TCP transmission protocol, it is more acceptable for applications that rely on the UDP protocol and thus need to handle packet reordering on their own, usually by data buffering.

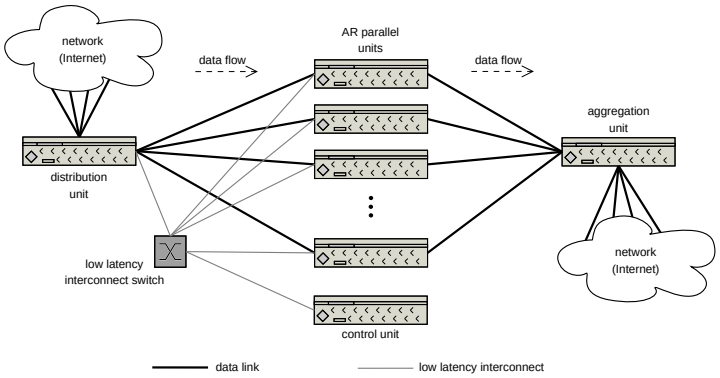

Fig. 1. Proposed VM-ready distributed router architecture.

In particular, because of the mentioned packet reordering problem, the transmission protocol called *Active Router Transmission Protocol* (ARTP, [2])[2] we previously designed is used as a transport protocol for our distributed active router.

2.1 Router Architecture

Distributed AR architecture we propose assumes the infrastructure as shown in Figure 1. The computing nodes form a computer cluster interconnected with each node having two connections:

- one *low-latency control connection* used for internal communication and synchronization inside the distributed active router,
- at least one[3] *data connection* used for receiving and sending data.

The low latency interconnection is necessary since current common network interfaces like Gigabit Ethernet or 10 Gigabit Ethernet provide large bandwidth, but the latency of the transmission is still in order of hundreds of μs, which is not suitable for fast synchronization of router units. Thus, the use of specialized low-latency interconnects like Myrinet network providing as low latency as 10 μs (and even less, if you consider e.g., InfiniBand with 4 μs), which is close to message passing between threads on a single computer, is very desirable.

From the high-level perspective of operation, the incoming data are first distributed across the multiple parallel units of the distributed AR, processed in these units, and finally aggregated and sent over the network to the next

[2] The ARTP is a connection-oriented transport protocol providing reliable duplex communication channel without ensuring that the data will be received in the same order as they were sent.

[3] The ingress data connection could be the same as the egress one.

node (or to the receiver). As obvious from the Figure 1, the router architecture comprises four major parts:

- **Distribution unit**—this unit takes care of ingress data flow distribution over multiple parallel AR units. The operation of the distribution unit depends on the case the router is used for. In the simplest scenario, when the distributed router is used for increasing the number of simultaneously running active sessions[4] only, it simply forwards all the incoming data to the proper and always the same AR unit. Considering more complex scenario, the distribution unit redistributes the incoming data to two or more AR units, which enables higher amount of data to be processed on the router. The data redistribution may be done using simple round robin fashion, but it may also support more advanced schemes like static and dynamic load balancing.
- **Parallel AR units**—the parallel AR unit receives data from the distribution unit and assembles them using ARTP protocol mechanisms into datagrams, which are forwarded to proper active session for processing. The processed data are fragmented into ARTP packets again and forwarded to the aggregation unit.

  Each AR unit is able to communicate with the other ones using the low-latency interconnection. Besides the load balancing and fail over purposes this interconnection is used for sending control information of active sessions (e.g., state sharing, synchronization), because each ARTP datagram may consist of control information which has to be forwarded to all AR units processing given active session. Thus, when such information is received, it is immediately forwarded to the distribution unit, which forwards it to the rest of AR units processing given session.
- **Control unit**—the control unit is responsible for the whole router management and communication with its neighborhood including communication with router users to negotiate new active sessions establishment and, if requested, providing feedback about their behavior.

  Concerning the router internal communication, the control unit directly communicates with the resource management module of each parallel AR unit via the low-latency interconnection and manages all the router resources available. For example, using the information about the router resources available the control unit decides, whether the new active session requests will be satisfied or not, or when and which active session will be migrated [3] to another parallel AR unit in order to effectively utilize all the router resources.
- **Aggregation unit**—the aggregation unit aggregates the resulting traffic to the output network line(s). Although the ARTP protocol used for active sessions data transmission is not as sensitive to packet reordering as the TCP protocol is, it is very desirable to design and implement a synchronized sending strategy to avoid large packet reordering with possible negative impact on further processing on the next node.

[4] Details about active sessions are given in Section 3.1.

3 Parallel AR Unit Architecture

Since the AR parallel unit could be seen as an independent active router, for our AR unit architecture we use a generic model of active router with loadable functionality proposed in [4]. Because of its modular architecture, we slightly modified the scheme in order to facilitate implementation based on virtual machines (VM).

The VM approach enables users of proposed router not only to upload the active programs, which run inside some virtual machine, but they are allowed to upload the whole virtual machine with its operating system and let their passing data being processed by their own set of active programs running inside uploaded VM. Similarly, the router administrator is able to run his own set of fixed virtual machines, each one with different operating system and generally with completely different functionality. Furthermore, the VM approach ensures strict separation of different virtual machines and also allows strict scheduling of resources to individual VMs, e.g., CPU, memory, and storage subsystem access.

The architecture of our VM-ready active router unit is shown in Figure 2. The bottom part is the VM-host layer where the core of the proposed AR unit is located. The core includes packet classifier, shared buffer pool, and packet scheduler modules. The modules relevant to the resource management (the resource management module and the VM/AP scheduler module) are described in more detail in Section 3.2. The Packet classifier module classifies all the incoming packets whether they belong to any active session running on the router. It also extracts packets destined to the session management module and sends them directly to that module. The shared buffer pool module operates as the buffer space where all the incoming packets are stored before further processing and there also are all the outgoing packets stored before the packet scheduler module sends them to the aggregation unit.

The VM-host management system has to manage the whole unit functionality including uploading, starting and destroying of the virtual machines, communication with a router control unit, and a session accounting and management. The virtual machines managed by the session management module could be either fixed, providing functionality given by a system administrator, or user-loadable. The example of the fixed virtual machine could be a virtual machine providing classical routing as shown in Figure 2. Besides that, the one another fixed virtual machine could be started as an active program execution environment where the active programs uploaded by users are executed. This virtual machine serves especially for backward compatibility with the original generic AR and this approach does not force users to upload the whole virtual machine in the case where active program uploading is sufficient.

3.1 Data Flow Through the AR Unit

The VM-ready AR unit architecture uses a connection-oriented approach similar to the one used in [4]. In terms of our architecture, the connection is also called *"(active) session"*, but with each active session consisting of one or more active

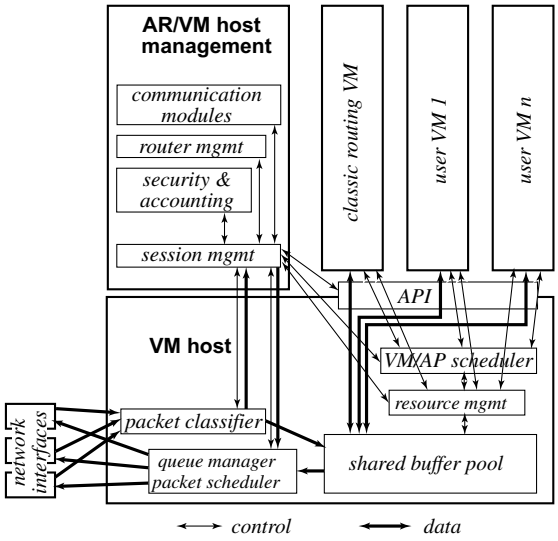

Fig. 2. VM-ready active router architecture

programs/virtual machines and one or more network flows. The association of more active programs/VMs and network flows into one active session is very useful especially when creating active programs working with more than one network stream (e.g., synchronization of two RTP streams when transmitting audio and video streams separately).

When a new active session request arrives to the router, the control unit decides, whether it could be satisfied or not. If the request could be satisfied, the session establishment takes place. Once the session is established, the data flow through the AR unit could be briefly described in the following way: when a packet arrives to a network interface, the packet classifier module decides, which active session the packet belongs to, based on an information from the session management module. Then, the classifier module forwards it to the appropriate VM/active program running on the AR unit or the establishment of a new session takes place. Finally, the active packet is processed in the VM and sent to the aggregation unit through the shared buffer pool.

3.2 Resource Management in VM-ready AR Unit

As obvious from the VM-based AR unit architecture described above, there are two main modules relevant to the resource management: (1) the resource management module and (2) the VM/active program scheduler.

Resource management module. This module is responsible for the management of resources inside the AR unit. It implements the crucial resource management scheme with the following functionality:

– keeping all the information about the resources in the AR unit,
– providing necessary information to the control unit and the session management module,
– monitoring and logging the amount of resources used by each active session.

VM/active program scheduler module. This module schedules the execution of the applications and the transmission of the packets of each active session to the next node. It is present especially for the future purposes, since we plan to support complex QoS guarantees as described in [5]—it will implement scheduling algorithms for different classes of resources to enforce the active sessions allocations of the AR unit resources.

4 Related Work

Thanks to lots of possible applications, the active networks principles are very popular and thus researched by lots of research teams. Thus, various architectures of active routers have been proposed—in this section we briefly describe only those ones mostly related to our work.

C&C Research Laboratories propose the CLARA—the prototype of a routing node in their JOURNEY network. The CLARA (CLuster-based Active Router Architecture) [6] consists of a cluster of generic PCs connected by a fast System Area Network (the prototype implementation uses Myrinet network) providing customizing of media streams to the needs of their clients. The CLARA provides fixed functionality only and does not guarantee the processing of all packets sent—additional guarantees must be implemented end-to-end, according to the requirements of individual streams.

The LARA (Lancaster Active Router Architecture) [7] architecture encompasses both hardware and software active router design. The LARA++ (Lancaster's 2nd-generation Active Router Architecture) [8], as the name indicates, evolved from the LARA. Against the LARA, which provided innovative hardware architecture, the LARA++ lays the main focus on the software design of the active router architecture—its software architecture is designed to be largely independent of the underlying hardware and thus, it could run on a single-processor node as well as use a distributed architecture. However, both router architectures do not provide user-controlled arbitrary active programs uploading.

5 Conclusions and Future Work

In this paper, we have proposed a virtual machine oriented distributed active router architecture. The main features of our router architecture are that it provides simultaneous parallel processing of passing user data using a cluster-based system with the ability of uploading user own code. Furthermore, we believe that the possibilities of uploading the whole operating system with their own set of programs as well as various operating systems running on the router and providing a fixed functionality are also very useful.

Concerning the future challenges, the proposed router architecture will be implemented based on the Xen virtual machine monitor [9]. Further we want to explore the mechanisms of QoS requirements assurance and their distribution over the AR parallel units. For the efficiency purposes, another interesting topic for our future work is the implementation of some parts of the router (especially the packet classifier module of each AR unit, distribution unit and aggregation unit) in hardware, e.g., based on FPGA-based programmable hardware cards [10].

Acknowledgments

This project has been supported by research intents "Integrated Approach to Education of PhD Students in the Area of Parallel and Distributed Systems" (No. 102/05/H050) and "Optical Network of National Research and Its New Applications" (MŠM 6383917201).

References

1. Smith, J.E., Nair, R.: Virtual Machines: Versatile Platforms for Systems and Processes. Elsevier Inc. (2005)
2. Rebok, T.: Active Router Communication Layer. Technical Report 11/2004, CESNET (2004)
3. Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live Migration of Virtual Machines. In: Proceedings of the 2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI), Boston, MA (2005) 273–286
4. Hladká, E., Salvet, Z.: An Active Network Architecture: Distributed Computer or Transport Medium. In Lorenz, P., ed.: Networking – ICN 2001: First International Conference Colmar, France, July 9-13, 2001, Proceedings, Part II. Volume 2094 of Lecture Notes in Computer Science., Heidelberg, Springer-Verlag (2001) 612–619
5. Rebok, T., Holub, P., Hladká, E.: Quality of Service Oriented Active Routers Design. In: MIPRO 2006 / Hypermedia and Grid Systems, Opatija, Croatia (2006) 206–211
6. Welling, G., Ott, M., Mathur, S.: A cluster-based active router architecture. IEEE Micro **21**(1) (2001) 16–25
7. Cardoe, R., Finney, J., Scott, A.C., Shepherd, D.: Lara: A prototype system for supporting high performance active networking. In: IWAN 1999. (1999) 117–131
8. Schmid, S.: LARA++ Design Specification (2000) Lancaster University DMRG Internal Report, MPG-00-03, January 2000.
9. Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Pratt, I., Warfield, A., m, P.B., Neugebauer, R.: Xen and the Art of Virtualization. In: Proceedings of the ACM Symposium on Operating Systems Principles, Bolton Landing, NY, USA (2003)
10. Novotný, J., Fučík, O., Antoš, D.: Project of IPv6 Router with FPGA Hardware Accelerator. In Cheung, P.Y., Constantinides, G.A., de Sousa, J.T., eds.: Field-Programmable Logic and Applications, 13th International Conference FPL 2003. Volume 2778., Springer Verlag (2003) 964–967