User-Empowered Programmable Network Support for Collaborative Environment

Eva Hladká[1], Petr Holub[1,2], and Jiří Denemark[1]

[1] Faculty of Informatics
[2] Institute of Computer Science,
Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic
eva@fi.muni.cz, hopet@ics.muni.cz, xdenemar@fi.muni.cz

Abstract. We introduce a user-empowered UDP packet reflector to create virtual multicasting environments as an overlay on top of current unicast networks. The end-users' ability to fully control this environment by a specific communication protocol is the main advantage of our approach. Serializing the parallel communication schema for group communication allows us to introduce special features that are possible in unicast communication only. Similar to working with programmable routers, users can submit their own modules, which can be linked into the reflector and perform user-specific operations (filtering, transcoding etc.). The reflector is the basic element of the overlay network support for the user-empowered group communication in collaborative environments.

1 Introduction and Theoretical Background

In the current world, people are looking for systems supporting easy-to-use and inexpensive activities like video-seminars, tele- and video-consulting, and virtual meetings, which are specific forms of a virtual collaborative environment [1]. This paper focuses on both building a theoretical framework and creating a practical implementation of a network support system for communication among smaller groups of participants (up to 20 sites, usually bellow 10) that can be fully controlled by the participants themselves. The system is intended to be simple to use and yet flexible, capable of reacting to pre-defined as well as dynamic events such as changes in number and location of participants, network conditions (bandwidth, delay, security), etc.

Two basic principles are being adopted in a complementary way when transferring data over the networks: *connection oriented* and *packet oriented* approaches. Both approaches reached widespread use in different environments and nowadays we see a lot of effort dedicated to their convergence. This is also dictated by new applications and their requirements of scalability on one hand and transport quality control on the other hand. The packet based networks with rather "dumb" active elements targeted for only one function—data routing—won the field of high-speed networks, while sacrificing most of the control features needed for advanced applications. A quality of service is offered on a statistical basis only (e. g. using DiffServ approach) and the users usually have no way of influencing or at least monitoring the transport of "their" data over the network. As reaction to these problems, we are developing a novel approach based on following cornerstones:

- *Active elements* within the network, programmable directly or at least indirectly by the users and their applications. These serve as the underlying technology for implementing the higher layers [2].
- *Overlay networks* as a framework for introducing specific services within the packet oriented network. The overlay networks allow minimizing the necessary overhead for advanced services without limiting their complexity [3].
- *User empowered approach* as a way to put the control plane into the hands of end users. The users can set up and tear down services, control and monitor their behavior while the services are well isolated so as to avoid any unwanted influence on other users.

The reflector is built as a special active node within a network, with full control by the user who uses it for group communication. The active router was modified to serve as the user controllable (user empowered) multimedia data reflector. The active node is implemented as a specific service within an ordinary computer. Fulfilling the requirement of full user control means overworking the active router and moving its functionality into the user space without any changes on the kernel level. This special implementation of an active router in user space was created and became the basic element for the overlay network for group communication.

2 Reflector

The *reflector* is a network element that replicates and optionally processes incoming data usually in the form of UDP datagrams and distributes this data to its clients in sequential manner using unicast communication only. If the data is sent to all listening clients, the number of data copies is equal to the number of clients. Our reflector is designed as a user-controlled modular programmable router, which can optionally be linked to special processing modules in run-time. The reflector runs entirely in user-space of the underlying operating system and thus it works without the need for administrative privileges on the host computer. The reflector architecture comprising the administrative part and data routing and the processing part is shown in Fig. 1.

The data processing and replication works as follows: the entry points of the reflector are network listener modules which are bound to one UDP port each. The received packet is placed into the shared memory and the listener adds a reference to a "to-be-processed" queue. A packet classifier reads the packets from this queue, checks with a routing AAA module whether the packet is allowed or not and determines a path through processor modules for each packet. After the processing, the data is distributed to clients by a packet scheduler/sender module according to a distribution list obtained from a session management module. The

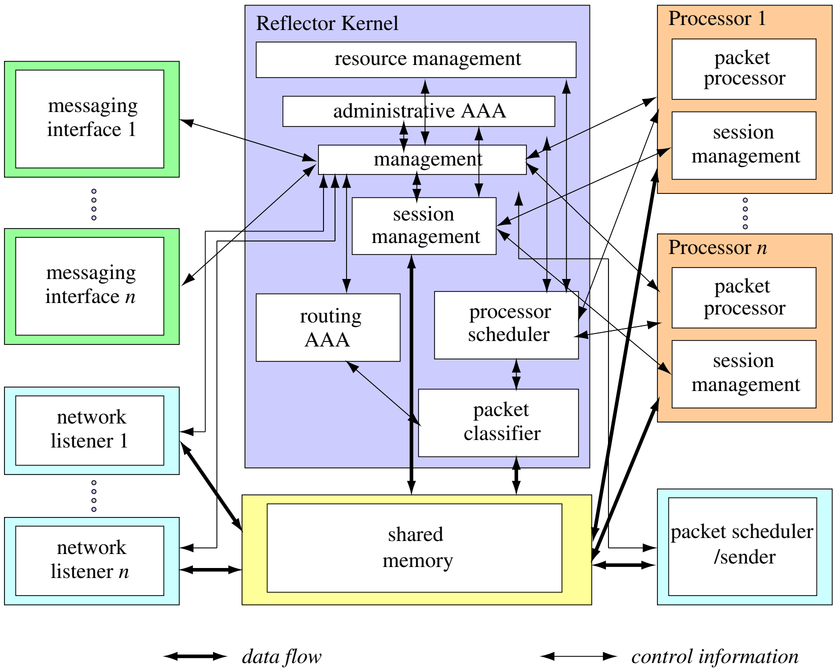

Fig. 1. Architecture of the reflector.

number of copies of the data inside the reflector is minimized in order to boost performance; for simple scenarios the reflector works in the zero-copy mode.

The session management module is responsible for maintaining the distribution lists for each group, for adding new clients (usually after the client starts to send data), and removing inactive (dormant) clients. Simple client authorization is based on IP address restrictions. The access control list contains an “accept” or “deny” rule for each IP address or subnet record. The decision is made by the routing AAA module, rejected packets are dropped and an appropriate event is generated and can be logged if requested.

The administrative part of the reflector can be accessed via secure messaging channels such as HTTP with SSL/TLS encrypted transport or SOAP with GSI support[3]. The user can authenticate using various authentication procedures, e. g. combination of login and password, Kerberos ticket, or X.509 certificate. Authorization uses access control lists (ACLs) and is performed on per-command basis. Authentication, authorization, and accounting for the administrative part of the reflector is provided by an administrative AAA module. The actual reflector control is provided by a management module, which accepts commands in a specific messaging language, the Reflector Administration Protocol (RAP) [4].

[3] Basic transport used for secure web services in Grid environments.
`http://www.doesciencegrid.org/Grid/projects/soap/`

All the events that occur in the reflector (users joining or leaving the reflector, exceptions etc.) can be logged for further inspection.

The data received by the reflector are replicated and sent back to all the connected clients and thus the limiting outbound traffic on the reflector grows with $n(n-1)$ where n is the number of active (sending) clients. The scalability issue arises obviously which can be mitigated by creating networks of reflectors with tunnels connecting them (see Sec. 3.1). The network can be built in either a static way (pre-configured) or dynamic way (e. g. using distance-vector routing algorithms or some more efficient routing algorithms from peer-to-peer networks like Pastry [5]). Reflector networks can also be used for building overlay networks that are more resilient to network outages than the underlying network [3].

2.1 Advanced Reflector Scenarios

Because of the data replication for each individual client, it is possible to implement per-user processing which is impossible to do with multicast. The modularity of the reflector allows users to add and configure specific functionality in run-time. Examples of per-user processing are shown below:

– *Multimedia data transcoding.* Data processing modules can convert data between different formats (e. g. re-compress data from the DV format to the H.261 format). The reflector can be thus used as a gateway allowing clients with limited support of compression formats or insufficient network or processing capacity to join videoconference without forcing the rest of the communicating group to use low-quality or low-bandwidth multimedia formats.
– *Video image composition.* Composing several video images into a single image can be useful for a collaborative environment with a large number of participants in which there is not sufficient processing or display capacity to provide full video windows from all the clients simultaneously.
– *Synchronization.* When using parallel media streams encapsulated in RTP protocol, it is possible to synchronize such streams [6]. RTP packets contain relative time-stamps that can be converted to absolute local time on the sending machine by utilizing both relative and absolute time-stamps sent in complementary RTCP packets. When the synchronized streams originate on different computers, it is necessary to synchronize time on these computers, e. g. using NTP protocol.

By connecting reflectors with different functionality, it is possible to create an overlay network allowing users to connect to reflectors according to their needs.

Reflectors can be used for building strongly secured communication and collaboration environments. In the secured scenario each client must maintain a secured reliable connection to the reflector (usually an SSL encrypted TCP connection) that is used to exchange encryption keys between the client and the reflector. UDP datagrams are then sent encrypted from the client to the reflector, processed, and distributed to other clients encrypted again. Such reflectors however, requires modified MBone Tools to work with [7]. The reflector can also be used in an adverse networking environment restricted by firewalls and NAT deployment since it is possible to tunnel UDP data between reflectors through a TCP connection using some well-known ports that are enabled on the firewall [8].

3 Performance Evaluation of a Prototype Implementation

The reflector described above has been implemented and its performance has been evaluated in order to verify its usability. The testbed environment comprised three powerful machines used as a traffic generator (`gerard`), a reflector (`test4`), and a receiver (`brand`). The machines were connected via the HP ProCurve 6108 gigabit Ethernet switch. More detailed information on configuration of these machines is shown in Tab. 1.

Table 1. Overview of configurations of the testbed machines.

	`test4`	`brand`	`gerard`
brand	–	DELL PowerEdge	DELL PowerEdge
model	–	1600 SC	1600 SC
processor	2× Intel Xeon 2.80 GHz	2× Intel Xeon 2.80 GHz	2× Intel Xeon 2.80 GHz
memory	1024 MB	1024 MB	1024 MB
NIC	Intel 82545EM 64 bit/66 MHz	Broadcom BCM5701 64 bit/100 MHz	Intel PRO/1000 32 bit/66 MHz
operating system	Linux 2.4.23	FreeBSD 5.2-RELEASE	FreeBSD 5.2-RELEASE

To evaluate the performance, clients sending 30 Mbps stream each were used thus emulating multimedia clients utilizing DV [9] video format sent in RTP packets over the IP network [10]. During the experiment the number of active (both sending and receiving) clients was increased and there was a single passive (listening only) client used as a measuring probe. The results summarized in Fig. 2 show that the system is usable for communication of up to five active clients working with very high quality video. For clients with lower bandwidth utilization, the number of clients that can get connected grows $n \propto 1/b$ where n is the maximum number of connected clients and b is the bandwidth used. It is also obvious from the results that the reflector is capable of fully saturating a gigabit Ethernet network link with limits imposed by the hardware and operating system used. The problem of scalability can be further tackled by building networks of reflectors (Sec. 3.1).

We have also evaluated the maximum forwarding throughput of the reflector which proves to be more demanding compared to common replication. This corresponds to the fact that more data is transmitted over the PCI bus compared to the replication mode. The results summarized in Fig. 3 show that it is possible to forward approximately 450 Mbps without significant packet loss.

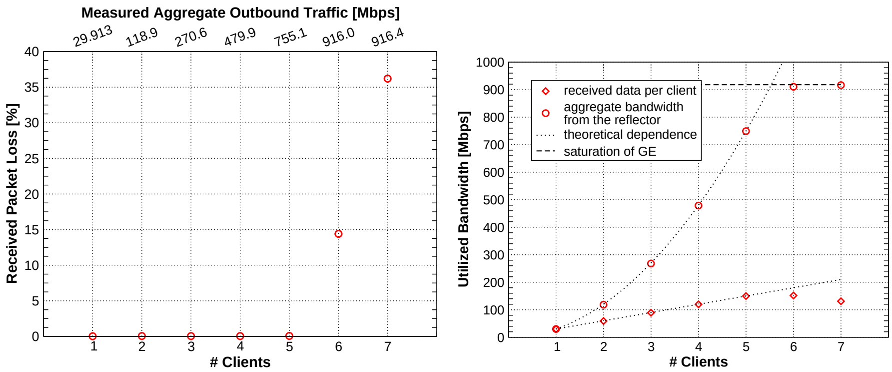

Fig. 2. Reflector prototype performance evaluation for 30 Mbps clients.

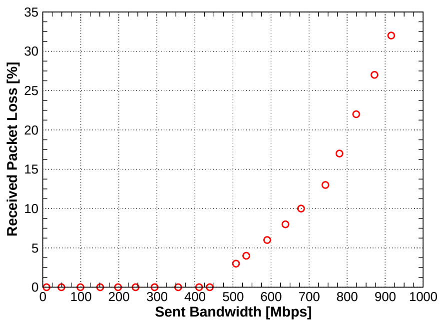

Fig. 3. Raw forwarding performance of the reflector.

3.1 Scalability Implications

As already mentioned in the Sec. 2 and 3, the scalability of the reflector-based communication environment can be further improved by creating networks of the reflectors connected via tunnels. The simplest model, which can also be used as the worst case estimate, is a complete graph in which each reflector communicates directly with all the remaining reflectors as shown in Fig. 4a. We call such model *full mesh tunneling*.

Let's assume a mesh of the reflectors in which each reflector has either n_r or $n_r - 1$ clients resulting in the most balanced population of reflectors with clients. It is possible to show that the number of inbound streams on each reflector is

$$in = n, \tag{1}$$

where n is the total number of clients. The number of outbound streams for reflector with n_r clients is

$$out_{n_r} = n_r(m + n - 2), \tag{2}$$

where m is the total number of reflectors in the mesh. The ratio of outbound traffic for the reflectors with n_r and $n_r - 1$ clients is

$$\frac{out_{n_r-1}}{out_{n_r}} = \frac{n_r - 1}{n_r}. \tag{3}$$

Taking into account that $n_r = \lceil \frac{n}{m} \rceil$ and the number of streams on a single stand-alone reflector is $out_s = n(n-1)$, it is possible to show that the ratio between the limiting outbound traffic for the reflector participating in the mesh of reflectors out_{n_r} and a stand-alone reflector out_s is

$$\frac{out_{n_r}}{out_s} \approx \frac{m+n-2}{m(n-1)}. \tag{4}$$

As illustrated in Fig. 4b for meshes with varying numbers of the reflectors, it is possible to increase the number of clients sending 30 Mbps to 15 when a mesh of 12 reflectors with gigabit network link is used.

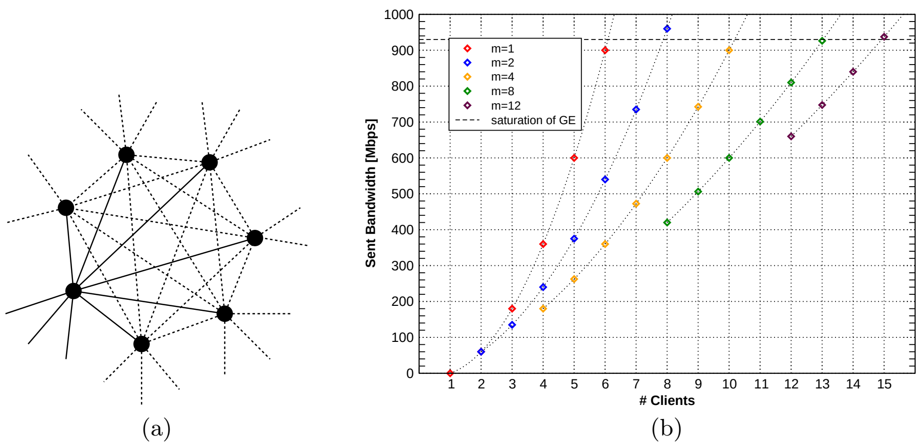

Fig. 4. (a) Full mesh reflector tunneling model. (b) Dependence of the number of 30 Mbps clients on the number of reflectors in the mesh.

4 End-User Applications

The reflector, capable of processing and distribution general purpose UDP packets, is able to support variety of end-user collaborative applications. Specific end-user requirements can be served by different versions, distinguished by specific processing capabilities (modules). Routinely, we use reflectors in connection with the `vic`, `rat`, and `wbd` tools from the MBone Tools package [11].

Different user groups used this environment during the last 3 years mostly for videoconferencing purposes, in very heterogeneous network conditions. While some groups enjoy rather homogeneous network environment, where all clients are connected through 100 Mbps network and the backbone runs on 1 Gbps and above speed, the reflector-based videoconferencing system is also regularly used in an environment where some clients are connected via cable TV. Even the network latency can be accommodated—we support a user group where most members are located in the Czech Republic (with clients both on high speed academic network and on a cable TV one) and one client connects via cable TV from Seattle (Washington, USA). Such an environment is rather hostile to the native MBone while reflector based data distribution works without problems. Another experimentally confirmed advantage is simultaneous support of different versions of MBone tools—again very problematic with native multicast.

As current high-quality videoconferencing tools tend to utilize high quality and high-bandwidth multimedia streams, we have also successfully tested the reflector with the DV over IP transmission tools from the DVTS project. We re-implemented the `xdwshow` tool (to overcome some problems encountered in its official implementation) and we have versions for PAL and NTSC formats under the Linux and FreeBSD operating systems. Our implementation uses robust thread architecture, where individual threads are used to input data from network, render them and display the resulting picture. This new implementation also support communication with reflector [12]. We plan to use this high quality video environment for teaching purposes, e.g. real time transmission from a neurological operation. Also, such high quality video streams can be used for the 3D video, using also the synchronization feature of the reflector.

5 Conclusion

The reflector is an active programmable network node providing all the necessary support for group communication in unicast networks. Our solution can simulate the multicast connectivity transparently, so the multicast clients can be used with ease, while keeping the advantages of unicast point-to-point communication lines. The reflectors function as multicast rendezvous points, allowing clients to connect and leave without any undesired influence on the rest of the group. The startup and shutdown of reflectors is a part of active network programming and as such it is fully user controlled. Users are also free to connect reflectors together in an *ad hoc* way and to specify behavior of each individual reflector, including possible security requirements and QoS parameters. A more general contribution is the method introducing new services into a network. The user empowered overlay network can be built a local scope (where needed), using only the features actually needed and within a limited time frame only. The network in this case is not overloaded by new protocols etc. and remains simple, robust, and fast.

While individual reflectors do not scale well and are able to support groups of tens of clients at most, their mesh is scalable enough to support a sufficient number of clients. Interesting direction of reflector development is the implementation of self-organizing and automatic discovery capabilities stemming from ideas of peer-to-peer networking. We will consider either the pure, hybrid or super-peer modes to define the best model for the reflectors self organization. The reflectors will be able to create overlay networks that can sustain even partial network disintegration without completely breaking overlay network.

As the reflectors can create the overlay networks, the reflector based solution does not depend on any specific functionality of the underlying network. All the advanced features are provided by higher user-controlled layers. Any group needing to collaborate can start its own reflector(s) and a unicast connectivity is the only required network capability from any client to the reflector. While the data routing and the data replication are automatically provided, user-specific services can be added as extensions (modules) to the reflector.

The future work will include control through grid service interfaces, as specified by an Open Grid Services Architecture framework [13]. This will enable easy integration with both next-generation collaborative Grid environments (e. g. AccessGrid version 2.x [14]) and with an optical network control plane, purposely built using the web/grid service approach. Thus the reflector will be able to cooperate easily with either the underlying network or other collaborative environment frameworks. The next direction in future work is concerned in development of user administration of reflector's data processing capabilities. Possible example of this administration is moderating data streams and creating communication environment for sub-group discussion inside the videoconferencing groups and fully moderated discussion like teaching in virtual classroom.

6 Acknowledgements

This research has been kindly supported by a research project "High Speed Research Network and its New Applications" (MŠM000000001) and "Optical Network of National Research and Its New Applications" (MŠM 6383917201). The authors would like to thank to Tomáš Rebok for helping with the implementation of performance evaluation tools.

References

1. Chin Jr., G., Myers, J., Hoyt, D.: Social networks in the virtual science laboratory. Communications of the ACM **45** (2002)
2. Psounis, K.: Active networks: Applications, security, safety and architectures. IEEE Communication Surveys (1999)
3. Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay networks. In: 18th ACM Symp. on Operating Systems Princpiles (SOSP), Banff, Canada (2001)
4. Denemark, J., Holub, P., Hladká, E.: RAP – Reflector Administration Protocol. Technical Report 9/2003, CESNET (2003)

5. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany (2001) 329–350
6. Rebok, T., Holub, P.: Synchronizing RTP Packet Reflector. Technical Report 7/2003, CESNET (2003)
7. Bouček, T.: Kryptografické zabezpečení videokonferencí. Master's thesis, Military Academy Brno (2002) Czech only.
8. Salvet, Z.: Enhanced UDP packet reflector for unfriendly environments. Technical Report 16/2001, CESNET (2001)
9. Internation Electrotechnical Commission: IEC 61834: Recording – Helical-scan digital video cassette recording system using 6,35 mm magnetic tape for consumer use (525-60, 625-50, 1125-60 and 1250-50 systems). (1998, 1999, 2001) Parts 1–10, `http://www.iec.ch`.
10. Ogawa, A., Kobayashi, K., Sugiura, K., Nakamura, O., Murai, J.: Design and implementation of DV based video over RTP. In: Packet Video Workshop 200. (2000) `http://www.sfc.wide.ad.jp/DVTS/pv2000/index.html`.
11. Hladká, E., Holub, P., Denemark, J.: Teleconferencing support for small groups. In: Terena Networking Conference '02, TERENA (2002) `http://www.terena.nl/tnc2002/proceedings.html`.
12. Hladká, E., Holub, P., Liška, M.: Modular communication reflector with dv transmission. In: VRS'04, PASNET (2004) Czech only.
13. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum (2002)
14. Childers, L., Disz, T., Olson, R., Papka, M.E., Stevens, R., Udeshi, T.: Access grid: Immersive group-to-group collaborative visualization. In: Proceedings of Immersive Projection Technology, Ames, Iowa (2000)