Video transcoding and rerouting in Forwarding on Gates networks

Thomas Volkert, Florian Liers
Technische Universität Ilmenau
Integrated Communication Systems Group
Ilmenau, Germany
{thomas.volkert | florian.liers}@tu-ilmenau.de

I. INTRODUCTION

A recent report from Cisco [1] shows that currently 40% of consumer Internet traffic is video. Moreover, Cisco forecasts that until the end of 2015 even 62% of the overall traffic will be video. As a result of the growing popularity of video applications, network providers suffer from the increasing multimedia traffic. Especially on the last mile, the available bandwidth is currently limited and not sufficient for the desired amount of multimedia streams. Moreover, current data networks are based on the Internet Protocol IP, which transports user data based on best-effort communication. Consequently, packets are dropped or delayed in an unpredictable manner if a network is highly loaded. These issues lead to artifacts in live video playback and stalling of on-demand video playbacks. To counter these problems, a network can try to route the data over a different path, providing higher bandwidth, or reduce the video data rate via transcoding. Such a controlled degradation can improve the overall quality of experience at receiver side.

With today's networks, a transcoding function between two application instances are commonly implemented on end hosts. To implement transcoding within a current network, a special application proxy is required in the network. Such a proxy has to be addressable at sender side and, therefore, it has to be visible. Additionally, the proxy application has to know the address of the destination application to be able to forward each transcoded frame to the original receiver. A more sophisticated approach for in-network transcoding would be an automatically placed function, which is not visible to both sender and receiver.

Our network architecture "Forwarding on Gates" (FoG) [2] provides the demanded flexibility. FoG is able to place functions on end hosts and within networks. It instantiates functions depending on the requirements of each transmission. In this demo, we show a streaming use-case instantiating video transcoding automatically within the network. Moreover, we show FoG's ability to reroute data streams in order to react on link failures. This supports the reliability of video transmissions in FoG networks.

II. FORWARDING ON GATES

FoG constructs networks out of functional blocks, called *gates*. Gates represent functions for packet processing provided by networks and end hosts. A gate can implement any kind of functionality. For example, a gate can represent classical network functions such as transmitting a packet between neighbor nodes, e.g. via Ethernet. Moreover, a gate can represent application-related functions such as video decoding and transcoding. Additionally, gates can have attributes describing their provided throughput rate and the expected delay for packet processing. Connections are implemented by chaining gates according to the connection requirements. Depending on the requirements and the chained gates, a FoG connection can represent a UDP or a TCP-like transmission.

The FoG architecture consists of the three logical components: routing, forwarding and authentication. The routing component calculates chains of gates required for a FoG connection. It uses a graph describing the available gates and the requirements of a connection as basis for route calculations. The forwarding component relays packets through gates according to given routes from the routing component. Furthermore, it reports available gates to the routing component. The authentication component secures the signaling and provides the basis for accounting.

The FoG forwarding uses an adapted index-based forwarding comparable to [3]. However, an index in FoG can represent any type of functionality. Moreover, a FoG route is constructed out of segments, which allows combinations of explicitly given routes and addresses defining intermediate FoG nodes. The entire end-to-end chain of functions is constructed incrementally by FoG's routing process based on these intermediate addresses. This new route structure and the incremental routing process are two of the main differences between FoG and other known approaches. Both improve the scalability by allowing the movement of states and limiting the local knowledge in routing instances.

To provide a maximum of scalability, FoG's routing is distributed over multiple autonomous routing instances. Each FoG node has to know at least one routing instance, which knows its current physical neighborhood including available gates. Based on this information, a routing instance is able to respond to route requests in its local surrounding. The inter-domain routing can either be done via flat BGP routing or via our hierarchical approach [4]. Since the overall routing knows about the available gates, it is able to detect if additional gates are needed in order to satisfy requirements. In this case, the routing instance requests new gates from the forwarding instance in order to use them for its route calculation.

In case of link failures, FoG is able to reroute ongoing transmissions automatically. This feature is implemented based on local rerouting schemes [5] enabling a fast recovery.

This work was funded by the German Federal Ministry of Education and Research under the project "G-Lab_FoG" (code 01BK0935). The project is part of the German Lab (http://www.german-lab.de) research initiative.

III. Demonstration Setup

Our demo shows the video streaming use-case depicted in Figure 1. The setup consists of three FoG nodes forming a FoG network without using IP. The application on host A sends a video stream and the application on host B receives it. For the video stream generation, we use our video conferencing application "Homer" [6]. It allows the streaming of a real-time video, which is either grabbed from a webcam or read from a file. In order to define transmission requirements and to send the video stream to the receiver, the application interacts directly with the FoG network stack via the GAPI [7], an extended network API.

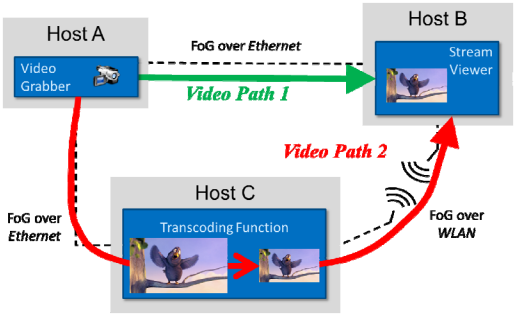

Figure 1: Network scenario with three FoG nodes

During the demo, the FoG protocol runs on all three hosts. It supports a dynamic setup of gates and concatenation of gates for the desired end-to-end transmission of video streams. Moreover, it supports gates interacting with hardware devices based on wired networks (using Ethernet) or wireless networks (using IEEE 802.11). This feature is used within the demo to perform the entire signaling for placing and managing functions.

From a user's point of view, the demo starts with a FoG-based video viewer on host B. The viewer opens a connection to the video source by specifying its name and requirements for the transmission. Based on the requirements, the FoG routing determines the gates required for the video transmission, e.g. a video decoding gate. Due to the available bandwidth on the wired link between host A and B, FoG uses existing unicast transmitting gates to send the stream from host A to B directly. The resulting route is shown as green arrow in Figure 1.

Figure 2 shows our GUI for FoG networks, visualizing the status of node B. The left hand side depicts the functional blocks residing on node B. In the middle, the received stream is shown in a real-time video viewer. On the right hand side, packets relayed by host B are shown. The video quality in the video viewer varies depending on the selected path and function chain within the FoG network. Additionally, this can be influenced by the user by unplug physically the cable between host A and B. FoG reacts on this link failure by initiating a rerouting of the video stream via host C. This alternative path is marked with a red arrow in Figure 1. It consists of both a wired and a wireless link. However, this wireless link provides a lower bandwidth than the green path. Consequently, packet loss can occur if the requested bandwidth remains as high as before. As mentioned in Section II, the forwarding of host C has informed its routing instance about the available gates and their capabilities. The routing is now able to decide if the wireless link is suitable for the transmission of the video stream. It compares the capabilities with the requirements of the transmission. If the bandwidth of the wireless link is not sufficient, the routing can trigger the creation of a transcoding gate automatically and uses this new gate in the route. In this case, the video quality on host B will be lower than before. However, stalling will be avoided. If the link between host A and B is available again, the FoG routing switches back to the old route.

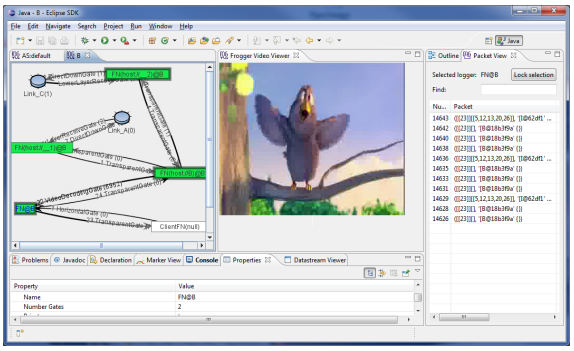

Figure 2: screenshot of the software prototype

To visualize the current routing, the GUI colors the used links for video streams. Additionally, the available bandwidth and the current data rates transmitted over links are shown.

IV. Summary and Conclusions

In this demonstration we show how our Future Internet architecture FoG can be used for automatic function placement within networks. We demonstrate this feature based on a video streaming use-case. Depending on the network capabilities, a transcoding function is instantiated and used within the network in order to provide the video even with a low end-to-end bandwidth. Moreover, FoG's ability to (re-)route data streams in order to fulfill transmission requirements is shown.

V. References

[1] Cisco Systems: "Cisco Visual Networking Index: Forecast and Methodology 2010–2015", white paper, 2011.

[2] F. Liers, T. Volkert, A. Mitschele-Thiel: "The Forwarding on Gates Architecture: Merging IntServ and DiffServ", 4th International Conference on Advances in Future Internet, Rome, Italy, August 2012.

[3] P. B. Godfey, I. Ganichev, S. Shenker, I. Stoica: "Pathlet Routing", In proceedings of SIGCOMM 2009, August 2009.

[4] T. Volkert, A. Mitschele-Thiel: "Hierarchical routing management for improving multimedia transmissions and QoE", IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM), San Francisco, USA, June 2012.

[5] R. Böringer, A. Mitschele-Thiel, G. Scharfe: "On Protection Schemes in MPLS-based Radio Access Networks", 13th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems (MMB 2006), Nürnberg, Germany, March 2006.

[6] T. Volkert: „Homer-Conferencing", web page: http://www.homer-conferencing.com.

[7] F. Liers et. al: GAPI: "A G-Lab Application-to-Network Interface", 11th Würzburg Workshop on IP: "Visions of Future Generation Networks" (EuroView) , Würzburg, Germany, August 2011.