A Media Streaming Protocol for Heterogeneous Wireless Networks

Antonios Argyriou and Vijay Madisetti
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA
Email: {anargyr,vkm}@ece.gatech.edu

Abstract— **In this paper we propose a protocol for media streaming in wireless IP-capable mobile multi-homed hosts. Next generation wireless networks like 3G, 802.11a WLAN, Bluetooth are the target underlying technologies for the proposed protocol. Operation at the transport layer guarantees TCP-friendliness, error resilience, and independence from the inner workings of the access technology. Furthermore, compatibility with UMTS and IMT-2000 wireless streaming service models are some of the additional benefits of our approach.**

I. Introduction

Second generation mobile systems offer primarily voice services and their enhanced versions restricted data services (e.g. GPRS). The next step, third generation mobile systems (3G) offer inherent support for IP-based data transfer. A number of media streaming standards have been currently proposed and standardized by the 3GPP group and the UMTS Forum for these systems [1]. These cellular-based technologies, are characterized by increased area coverage which is their biggest advantage. On the other hand, 802.11a/b wireless LAN technology (WLAN), offers high bandwidth connections at low cost but in a limited range. These two mainstream wireless access methods, have proponents that claim the dominance in the wireless broadband Internet market. However, the most probable scenario, wants the coexistence of both and even the addition of new technologies in the landscape (e.g. Bluetooth, HiperLAN). So the future landscape that formulates wants multiple, heterogeneous networks with overlapping coverage and operated by a number of competing service providers. The need for mobile terminals that will be able to access all these heterogeneous networks is unavoidable, making thus common the phenomenon of multi-homed hosts with WLAN, 3G, Bluetooth or other interfaces.

Even though wireless service providers are still in quest for killer applications, wireless video streaming is expected to be the most important application in IP-capable third generation (3G) wireless cellular networks. A number of protocols and inter-networking architectures will be needed for achieving seamless service provisioning in this highly diverse wireless environment. The incompatibility of the aforementioned access technologies makes problematic the inter-operation and results in Quality of Service (QoS) problems. These problems arise when the user wants to switch from one technology to the other that may either be priced differently or provide better service or are available in a place where another one is not. The possibility of using a number of access technologies concurrently for improved throughput must also be considered.

Numerous approachers have been proposed throughout the last years for video streaming in wireless systems. Application layer techniques dominate the proposals for streaming media over 3G networks and are based either on optimized implementations of application layer protocols RTSP/RTCP/RTP [2] or in extensions of well-known resource reservation mechanisms (RSVP) initially designed for wireline systems [3]. PacketVideo also provides a streaming platform [4] that is based primarily in hardware specific, optimized codec implementations (MPEG-4), and proprietary error-concealment techniques. Other approaches include error resilient and layered video coding for reducing the effect of packet losses in the perceived video quality [5].

In this paper we are concerned with a different issue in the video streaming process: How we can efficiently use the available bandwidth from all the wireless interfaces of the mobile host in order to provide higher throughput for the video streaming application. Our efforts led to the development of MSCTP (Media SCTP): a media streaming protocol that is based on the Stream Control Transmission Protocol (SCTP) [6], [7] and is particularly suited for multi-homed hosts and streaming servers that participate in end-to-end video streaming applications.

The rest of this paper is organized as follows: Section II provides an overview of the protocol functionality. Sections IV and V provide details about the algorithms employed by the protocol. Simulation results are presented in section VI, while section VII concludes the paper.

II. System Overview

The main problem addressed by our protocol, is the concurrent operation of multiple wireless interfaces in a seamless manner allowing thus bandwidth aggregation for a single media streaming session. This goal is achieved, because our protocol operates at the transport layer (assuming an IP-based underlying capable network) allowing thus to handle everything in an end-to-end fashion. Careful media stream scheduling/dispatching over destination interfaces at the server, allows for maximum resource utilization and performance.

Additional benefits such as the compliance with the currently standardized SCTP, and TCP-friendliness allow smooth inter-operation with currently deployed Internet infrastructure. Moreover, the higher layer nature of the protocol makes it compatible with streaming service models that have been proposed for 3G systems.

The protocol can ideally handle any number of interfaces available in the mobile host. For example in the case where a mobile host is reachable from another WLAN base station or a cellular network, the user will be able to initiate a concurrent connection with it (Fig. 1). Assuming that admission control fails, the user is notified about the fact that it may be possible a disruption of its current session if he continues roaming. In the opposite case, the MSCTP protocol will seamlessly use both the interfaces. One important assumption that we make is the following: The physical paths that correspond to the two interfaces do not overlap. This assumption is necessary so that we avoid shared congestion points between the subflows of the main media flow. Another important function of our protocol, is that it employs intelligent media packet scheduling at the transport layer. Goal of the scheduler is, as always, a timely delivery of media packets to the client so that the playout scheduler at the client does nor starve. Overall the functions that our protocol implements are : 1) Monitoring network status 2) dynamic mapping of a user level media streams to outgoing interfaces 3) high level object based adaptation at the transport layer.

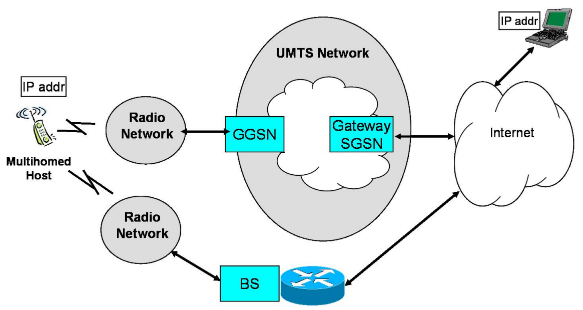


Fig. 1. Heterogeneous wireless network

A. Network model

We use a simple but widely used network model, defined in [8], so that is simple enough for being practically applicable. Further elaboration of our model is part of our future work. Our network model assumes the existence of a number of user data flows. These streams require concurrent transmission over the set of communication links which their number is L. The capacity of each link is denoted as C_l with $l \in L$. Let also λ_s denote the bandwidth allocation for stream s. All feasible bandwidth allocations must respect the following equations:

$$\sum_{r=l} \lambda_r \leq C_l \tag{1}$$

$$\sum_{l=1}^{L} C_l \leq C_{total} \tag{2}$$

III. Monitoring path status

Precise monitoring of the network status of each outgoing path, is crucial for making effective media packet scheduling decisions. Additional care must be paid so that network probing is reduced to a minimum and so bandwidth is not wasted. For this reason, our protocol maintains monitoring information that is already available at the SCTP stack. The metrics that we are of interest to us are:

Latency: Latency is usually measured by the use of Round Trip Time (RTT). RTT is alre ady maintained for each one of the transport addresses of an SCTP association.

Bandwidth: Maintaining information about the available bandwidth of a specific path, is crucial. The formula we use to estimate the network bandwidth available for a specific path is well known [5] :

$$\lambda = \frac{1.22 \times MTU}{RTT \times \sqrt{p}} \tag{3}$$

Estimating the available bandwidth is this way, guarantees that our algorithm will get bandwidth in a TCP-friendly fashion. The available bandwidth is recalculated every time we have a change in the value of either RTT or MTU. Note that this equation, has been shown to be valid for relatively long lived session which is the case for streaming applications.

Jitter: Variation in delay is of primary importance for interactive and streaming media applications. This class of applications require very low jitter. A variable $path\_jitter$ is maintained for each outgoing path and recalculated every RTT.

Packet Loss Rate: Increased packet loss rate for a channel, means that it is not reliable enough. The $loss\_rate$ variable is used for keeping track of the packet loss rate for each path and estimate the available bandwidth from equation 3.

IV. Packet scheduling algorithm

The scheduling algorithm dispatches RTP packets from userland that belong to various media flows to interfaces and thus to physical network paths. Our scheduling algorithm controls this process in an intelligent fashion so that it maximizes network resource utilization (i.e path bandwidth). Let us first give a brief example that depicts the rationale behind our algorithm. Suppose that a mobile client initiates a media streaming session form a server. Assuming that a connection has been established, the protocol at the server side, examines the state of each path with the client periodically and calculates the metrics described previously. Its purpose is to make sure that at all times, the media streams are allocated a specific part of the total bandwidth so that an overall metric function is optimized.

By having precise knowledge of the available paths between the two endpoints, and their corresponding status, the protocol assigns a $label$ to each one of the paths. This kind of $label$

exists for each one of the monitored metrics mentioned in the previous section. Primary purpose of this approach is to create a lightweight mechanism for maintaining the relative "quality" of each path according to our metrics. Following the above procedure, each of the user streams in an MSCTP session is mapped initially to one these paths. This means that all media data from this stream will be sent though its correspondent path. Additionally each user stream will have to provide a $stream\_qos$ variable that indicates its current required QoS. With this method, specific user streams can be assigned different priorities. In this way the stream that requires the best available service will correspond to a path that satisfies as close as possible these requirements. So basically the media packet scheduler implements a bandwidth allocation mechanism based on a weights optimization function [8]. Given that all the media streams are S then for each stream $s \in S$ there is at least one link $l \in L$ such that:

$$\frac{\lambda_s}{\phi_s} = \max\{\frac{\lambda_{s'}}{\phi_{s'}} : s' \in l\} \qquad \text{with} \qquad \sum_{s' \in l} \lambda_{s'} \leq C_l \tag{4}$$

Each stream is allocated a bandwidth λ_s according to its specific weight factor ϕ_s. The protocol increases bandwidth allocation gradually with a step ϕ_s for the each stream s. In case were our streams require low delay paths, bandwidth allocation is be performed in a different way. The use of this method is an option can be provided through socket API parameters [6]. The minimum potential weight allocation is our criterion for bandwidth allocation in this case [8], and so the objective is to minimize $\sum \phi_s/\lambda_s$ (with $s \in S$), since $1/\lambda_s$ denotes the potential delay for stream s.

A. RTT mismatch issues

As it is obvious, media flow packets may be dispatched to many outgoing interfaces. As a result packets may follow different paths and eventually received out-of-order at the receiver. These packets are collected and buffered so that they will be delivered in an ordered fashion. In order to avoid the large possible mismatches in the reception time of successive packets, we defined a set of packet sending rules for our protocol. Generally, the round trip time (RTT) can be set equal to: $RTT_l = \frac{p}{b_l} + t_l$, where p is he packet size, b_l is the bandwidth of the link l, and t_l is the total delay of various network components (e.g. queuing delay, etc) [9]. Key point in the packet scheduling algorithm is the ratio between these two RTT components. If t dominates the overall RTT then the available bandwidth can be exploited better in both the links since variations in the p/b ratio are less important for the value of RTT. In the other case, we can see that we have control over the p_l/b ratio since we can modify the size of an outgoing packet. In this way we can have a rough control over the RTT, if of course the p_l/b ratio dominates the overall RTT. In order to keep the RTT of every link nearly the same we have to arrange p_l/b for every link to be nearly equal. This can be achieved by splitting the data that will be sent in each interface. This splitting can be achieved easily in SCTP through the DATA chunk mechanism [6]. This means that we will place data chunks in a packet until a reach the limit set by the ratio p_l/b is reached for a specific interface. So, if the total data available for transmission from the userland is D, then at each link must be sent chunks with data size D_l, where $D_l = (b_l/b) * D$ where b_l represent the available bandwidth for link l and b represents the total aggregate bandwidth for of all the links.

```
SEND_DATA(object o)
    for each interface i ∈ I[N]
        do
            if BW-address(i)≥size(o− >base_obj)
                then send-data(i)
                goto send-more
            else FRAGMENT-DATA()
    send-more:
    for each enhancement object j ∈ enh_obj[M]
        if BW-address(i)≥size(o− >enh_obj(j))
            then send-data(i)
            goto send-more;
        else continue
```

Fig. 2. Pseudo code that depicts media flow adaptation

V. Adaptation of media flows

The media packet scheduler at the sender employs a number of additional intelligent features: It can decide if and when it should drop redundant media data (not actually commit to the network layer) due to network congestion. However, in order for our system to be able to perform this "dropping" it must know which information is the "basic" that must be delivered and which part of the data can be dropped under network congestion or increased packet loss conditions. In order to capture this high level information, we have defined a high level object based abstraction for expressing application level semantic information. We implemented this distinction by a labelling mechanism that assigns labels to each stream that an application delivers to the transport layer. These labels must be in the form $base\_obj, enh\_obj(1), enh\_obj(2)$ etc. Fig. 2 pseudo code depicts this part of the protocol.

In this way the problem of network congestion, is avoided initially: We do not inject excessive network traffic and latter use a packet dropping mechanism in a router to drop the packets due to congestion. The algorithm decides according to the observed network traffic if and when to inject the supplementary media packets in a transparent manner to the application.

VI. Simulation results

In this section we present simulation results obtained from simulations through the use of ns-2 [10], [11]. We defined a simple network topology that is depicted in Fig. 3. The topology consists of a mobile host, two base stations, a router, and remote host that communicates with the mobile host. The respective link bandwidth, delays are shown in Fig. 3.

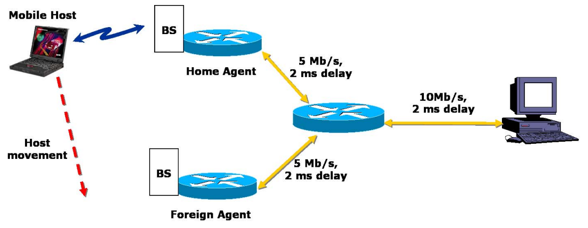


Fig. 3. Wireless streaming experiment setup

According to the simulation scenario, the client requests a media stream from a server. The server has encoded H.263++ VBR video and transmits video with the IP/UDP/RTP or with IP/MSCTP/RTP. In this case routers were configured with drop-tail policy as is usually the case in the Internet today. In order to measure video quality we used the Peak Signal to Noise Ratio (PSNR) of the luminance component [5]. We additionally developed simple packet loss models for wireless channels as defined in [12].

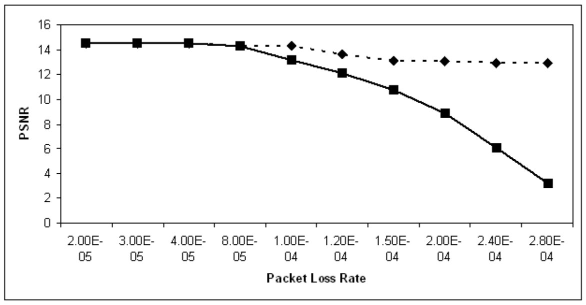
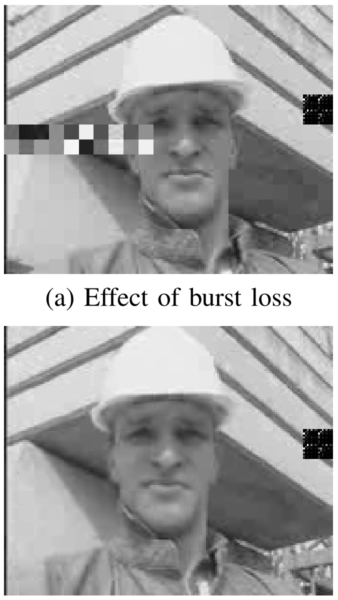


Fig. 4. PSNR vs. packet loss rate for MSCTP & UDP (video rate of 80kbps)

Fig. 4 presents PSNR results for various burst loss values. As we can see from this figure, after a specific threshold for the packet loss rate, the quality of the delivered video at the client drops dramatically. However, when both the client and the server used our system, we can see only slight decrease in the quality of the video at the client. Why this is happening? When MSCTP at the server has identified increase in the packet loss rate, it immediately starts streaming the video sequence from the other physical path from which the client is reachable. However, there is a slight decrease in PSNR even in this case and this is because the server suffers initially a specific, small packet loss in order to identify the start of the burst losses. After this has happened it switches to the other interface, and thus to different physical path. A small number of captured video frames after running the above experiment, can be seen in Fig. 5. Fig. 5(a) presents the frame when it has suffered a burst loss while being transmitted with UDP. However, in Fig. 5(b) the same frame suffers only a small burst loss when it was transmitted with MSCTP. This is because, as we said before, our system switches to another interface without allowing a burst loss to affect the specific frame even more severely.



(a) Effect of burst loss

(b) Preventing further packet loss

Fig. 5. The foreman video sequence

VII. Conclusions

In this paper we presented the modification of newly adopted transport layer protocol (SCTP) for streaming media over heterogeneous wireless networks. Its TCP friendliness is clearly depicted from our preliminary experimental results. Moreover, compatibility with UMTS and IMT-2000 IP-based streaming service models is guaranteed since the protocol operates at the transport layer.

References

[1] UMTS Forum, http://www.umts-forum.org.
[2] H. Montes *et al.*, "IP Multimedia Streaming Services in Third-Generation Mobile Networks," *IEEE Wireless Communications*, vol. 9, no. 5, October 2002.
[3] B. Moon *et al.*, "Reliable RSVP Path Reservation For Multimedia Communications under an ip micromobility scenario," *IEEE Wireless Communications*, October 2002.
[4] PackeVideo Corp. http://www.packetvideo.com.
[5] Y. Wang, J. Ostermann, and Y.-Q. Zhang, *Video Processing and Communications*. Prentice Hall, 2002.
[6] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, "Stream Control Transmission Protocol," RFC 2960, October 2000.
[7] R. Stewart *et al.*, "SCTP Partial Reliability Extension," draft-stewart-tsvwg-prsctp-00.txt, May 2002.
[8] L. Massoulie *et al.*, "Bandwidth Sharing: Objectives and Algorithms," *IEEE Transactions on Networking*, June 2002.
[9] D. S. Phatak and T. Goff, "A Novel Mechanism for Data Streaming Across Multiple IP Links for Improving Throughput and Reliability in Mobile Environments," in *IEEE INFOCOM*, New York, 2002. [Online]. Available: citeseer.nj.nec.com/481247.html
[10] S. McCanne and S. Floyd, "ns Network Simulator," http://www.isi.edu/nsnam/ns/, 1995.
[11] A. Caro *et al.*, "SCTP module for ns-2," http://www.pel.udel.edu.
[12] I. Stojmenovic, *Handbook of Wireless and Mobile Computing*. John Wiley and Sons, 2002.