Offloading cellular networks with Information-Centric Networking: the case of video streaming

Andrea Detti, Matteo Pomposini, Nicola Blefari-Melazzi, Stefano Salsano
CNIT- Electronic Engineering Dept.
University of Rome "Tor Vergata", Italy
e-mail: {name.surname}@uniroma2.it

Andrea Bragagnini
Telecom Italia, Turin, Italy
e-mail: andrea.bragagnini@telecomitalia.it

Abstract— **In this paper we present a cooperative video streaming application running on top of an "Information Centric Network" (ICN). The application could be used on mobile devices to offload the cellular radio interface. We demonstrate our application in a test-bed exploiting the CCNx software implementation of an ICN, the VideoLan tool and the Apple HTTP Live Streaming format.**

Keywords- Information Centric Network; video streming ; cellular network

I. INTRODUCTION

An Information Centric Network is a communication system specialized in distributing contents, rather than providing host-to-host connectivity. Each content is uniquely identified by means of a *name*, (e.g. a string). To fetch content, an user sends a content-request that includes the content name. On the base of this name, the ICN routes the content-request toward the best node that can provide the content and then sends back the content to the user.

The ICN concept is attracting more and more interest, (see e.g., papers [1][2][3][4]) and projects [5][6][7]). Among these proposals, Content-Centric-Network (CCN) [3], is one of the most known, thanks also to the development and release of an open source implementation called CCNx [8], for Linux, Mac and Android platforms. Within CCN, the nodes *route-by-name* content-requests and temporarily store forwarded contents in so-called *in-network caches*.

In this paper we propose a cooperative video streaming application for mobile cellular devices based on ICN concepts, which can be used to offload the radio cellular interface. The offloading is achieved by a suitable exploitation of the routing-by-name and in-network caching functionality of the ICN. We demonstrate our ICN application in a test-bed using the CCNx software, the VideoLan tool and the Apple HTTP Live Streaming format.

In our scenario we assume that a group of neighboring users use their mobile devices to view the same video, at the same time. The video is offered by a streaming server located in the fixed network. The mobile devices are ICN capable, they are connected to the fixed network by means of a cellular radio interface and connected to each other by means of a local wireless technology, e.g. a WiFi ad-hoc connection.

The video stream is structured in "segments"; each segment is a piece of the whole video (e.g. 10 sec) and has a unique ICN content-name. This streaming scheme is used by the Apple HTTP Live streaming protocol [9] and also by the forthcoming MPEG-DASH; in both cases, the unique content-name of a video-segment is a plain URL.

Thanks to our ICN application, a specific video segment is downloaded through the cellular interface only by a single user. Afterwards, that segment is shared among neighboring users by means of the WiFi ad-hoc network. This has the effect of offloading the cellular radio interface, if compared to a traditional HTTP streaming, where each user downloads all segments. The basic underlying idea is that *for each segment, only one device plays the role of local-gateway for the other neighboring devices*. All requests of a given segment are routed-by-name towards the local-gateway device, whose content-cache becomes the common en-route cache of the entire group. The content-cache serves all requests of a given segment via WiFi, except the first request, which is forwarded to and served by the origin streaming server, through the cellular interface.

To share the cost of the cellular connection in fair way among users, each device of the group plays the role of local-gateway for different sets of video segments, so as to equalize the number of segments downloaded over the cellular interface.

To the best of our knowledge, our ICN application is new; the aim of this paper is only to show it in a real test-bed and to propose a possible exploitation of the ICN paradigm in a cellular environment. We are not maintaining that our application has better performance, with respect to other "non-ICN" applications proposed in the literature, such as those based on network-coding [10]. Furthermore, we do not consider the presence of free-riders or misbehaving users, which could however be handled by filtering incoming content requests, e.g. adapting P2P strategies [11].

II. BACKGROUND

In this Section we briefly recall basic concepts regarding CCN and Apple HTTP Live streaming, useful to understand our demo.

CCN – In CCN, a content is divided in "chunks", each chunk has a unique name. A name has a hierarchical structure formed by Components, separated by a "/", i.e. "/Component-1/Component-2/…Component-#*n*". Usually,

978-1-4673-1239-4/12/$31.00 ©2012 IEEE

the first *k* Component represents the content-name, while the *n-k* remaining Components represent specific chunk information, such as version and chunk number. For instance, "/foo.eu/video001.ts/%03" could be the name of the chunk number 3 (%03) of a video whose content-name is "/foo.eu/video001.ts"

In order to fetch a content, a user sequentially downloads all its chunks. To request a chunk, the user sends to the network an *Interest* packet, which includes the chunk-name.

Each CCN node uses a name-based Forwarding Information Base (FIB) to relay Interest packets to the next CCN node of the path. This procedure is referred as *routing-by-name*. An entry of the FIB has the form <name-prefix, output-face>, where name-prefix is a sequence of Components and output-face specifies the means to reach the next CCN node, e.g. a UDP socket.

To forward an Interest message, the selection of the proper FIB entry follows a longest prefix matching policy, based on Components. For instance, considering an Interest for "/foo.eu/video001.ts/%03", the name-prefix "/foo.eu/video001.ts/" matches two components, while the name-prefix "/foo.eu/" matches only one component. If more FIB entries match the chunk-name included in the Interest message, the message is sequentially forwarded through all the output-faces, with a delay between consecutive forwarding operations.

During the forwarding operation, the node stores in the *Pending Interest Table* (PIT) the input face on which the Interest packet has been received. When the Interest packet reaches a node that has the content, the node sends back the content in the payload of a *Data* packet. The Data packet is routed back to the requesting user by using and "consuming" the state previously left in the PITs of the end-to-end path. Each CCN node has a content-cache to temporarily store forwarded Data packets.

Apple HTTP Live streaming – This solution enables sending audio and video over HTTP from an ordinary Web Server and supports both live broadcasts and pre-recorded content (video on demand). To view the video, the client (e.g., VLC) downloads a "m3u8" playlist file. The m3u8 file contains the list of the names (i.e. URLs) of the video segments that compose the stream and some TAGs, which support the timing of the streaming operation. After downloading the m3u8 file, the client starts to continuously "pre-fetch" a small set of video segments, so as to be ready to play them, remaining always slightly in advance with respect to the actual video playing. The client can pause or jump forward/backward in the sequence of video segments, following user commands. Currently, Apple HTTP Live streaming is supported by Apple devices, latest Android phones, and VLC 2.0 application for PC.

III. ICN COOPERATIVE VIDEO STREAMING

In this section we describe our ICN video Streaming Application (ICN-SA) using the example in Fig. 1

The streaming server, not reported in the figure, is located in the fixed network. It is an ICN content repository, which publishes all segments of a video, including its m3u8 file. The segments have the unique name /foo.eu/video00*x*.ts, where *x* is the segment number.

We consider the case of two neighboring ICN 3G mobile devices, named #A and #B. These devices have a preloaded FIB containing the default "/" entry towards the ICN gateway provided by the 3G network (3G-gateway), and the entry "/prd/", used for the *proximity-route-discovery* (pdr) operation, which uses the WiFi multicast channel 224.0.0.204.

We assume that both devices have already downloaded the m3u8 file and that user #A plays the video slightly in advance with respect to user #B.

The video client #A (e.g., VLC) requests the first segment: /foo.eu/video001.ts. The request is handled by ICN-SA #A, which starts a proximity-route-discovery procedure aimed at discovering if another device has been set as local-gateway of /foo.eu/video001.ts.

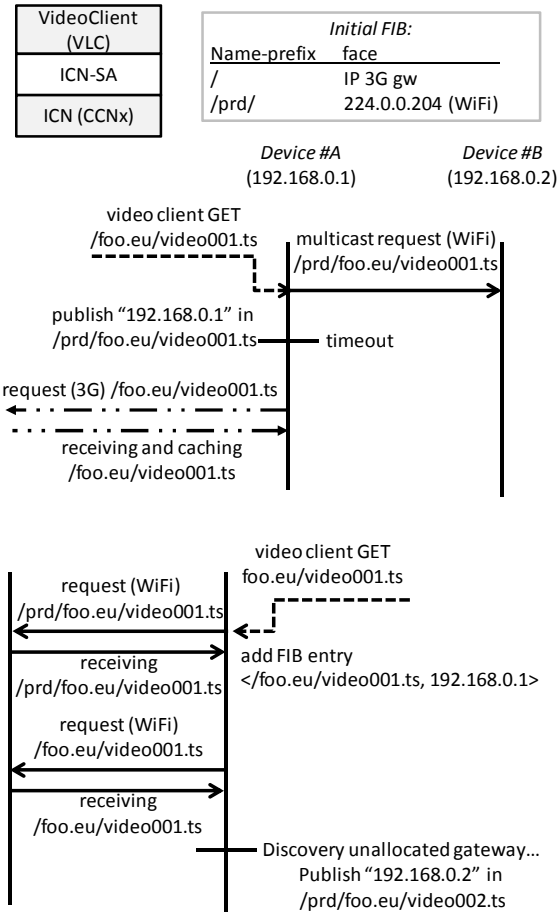

Fig. 1 - ICN cooperative video streaming for mobile cellular devices

We propose to use the ICN mechanisms also to support the proximity-route-discovery procedure. Indeed, when a device sets itself as a local gateway of the segment /foo.eu/video00*x*.ts, it publishes a *proximity-route-content*, named with the same name of the content to be routed (/foo.eu/video00*x*.ts), but with the special prefix "/prd/", i.e. /prd/foo.eu/video00*x*.ts. The payload of the proximity-route-

content contains the face that the local-gateway device is offering, e.g., its IP address 192.168.0.1.

To discover a proximity-route for /foo.eu/video001.ts, ICN-SA #A merely requests the related proximity-route-content /prd/foo.eu/video001.ts on the WiFi multicast interface. Since no other local-gateways has been elected for /foo.eu/video001.ts, the proximity-route-discovery fails after a timeout. Consequently, the ICN-SA #A configures the device #A as a local-gateway for /foo.eu/video001.ts, by publishing the related proximity-route-content. Then, the ICN-SA #A requests the content /foo.eu/video001.ts to the ICN, which routes-by-name this request, using the default FIB entry towards 3G-gateway. The content request is received from the video server, cached by the device #A and passed to the video client #A.

After few seconds, also video client #B requests /foo.eu/video001.ts. The ICN-SA #B performs the proximity-route-discovery procedure and obtains the proximity-route-content /prd/foo.eu/video001.ts. Using the payload of this content, the ICN-SA #B adds the route </foo.eu/video001.ts, 192.168.0.1> to its ICN FIB. Then, ICN-SA #B requests the content /foo.eu/video001.ts to the ICN, which routes the request to WiFi 192.168.0.1 (device #A), rather than to the 3G-gateway, owing to the longest prefix matching policy. The content cache of the device #A sends back the content via WiFi.

To fairly share the cost of the cellular connection, once that ICN-SA #B has received a segment, it has to offer itself as a possible local-gateway for other segments. To this end, it searches proximity-routes not yet offered by other devices; in our figure, it discovers that the proximity-route-content for the segment /foo.eu/video002.ts is not available, and sets itself as a local-gateway for this video segment, by publishing the related proximity-route-content /prd/foo.eu/video002.ts. These procedures are repeated for all other video segments.

IV. TEST-BED

The test-bed is composed of three Linux laptops (Fig. 2), playing the role of the video server and of the two mobile devices #A and #B. Mobile devices are connected to each other by a WiFi ad-hoc link and to the video server with a wired Ethernet link, in lieu of the real 3G connection (which might not be available on the demo premises).

Mobile users run VLC to view videos. VLC is configured so that the ICN-SA is its HTTP proxy. ICN-SA receives the HTTP GET requests issued by VLC and bridges these requests in the ICN domain, performing the procedures described in the previous section. When the ICN-SA receives a video segment from the ICN, it sends back the segment to the VLC, in an HTTP compliant format. All functionality of the streaming GUI, e.g. pause, forward, backward, etc. is supported.

To show how our ICN application can offload the cellular radio interface, with respect to a plain HTTP streaming, we setup on the video server device both an ICN content repository and a plain HTTP server. During the demo, we compare the cellular bandwidth consumed by using ICN and plain HTTP/TCP/IP. The plot is shown at run time on all devices, allowing to appreciate the changes in bandwidth usage deriving from interaction with the VLC GUI. In Fig. 2 we show an experiment performed in our lab. In the case of a single stream, ICN-SA uses more cellular bandwidth than HTTP, because of the security-related overhead included in every ICN data-unit (if we omit this overhead the results are equal to each other). In case of two video streams, ICN-SA consumes the same cellular bandwidth measured with a single ICN stream, while HTTP uses twice as much the bandwidth used for a single stream. This proves that we can offload the cellular interface. All software of the demo is available in [12].

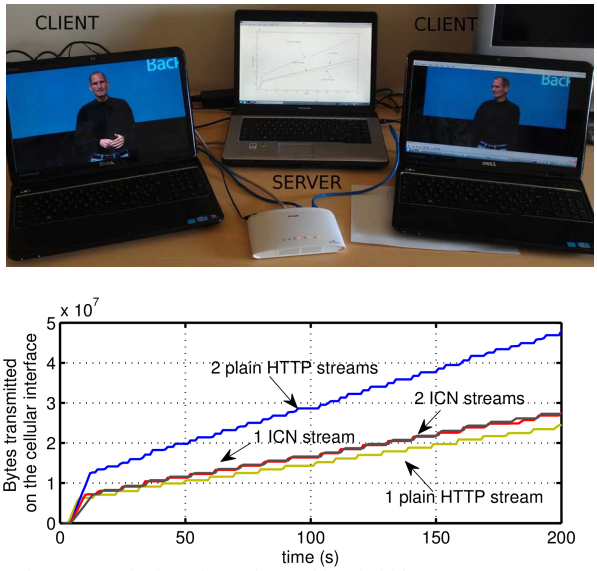

Fig. 2 – Test-bed configuration and bandwidth usage measurement

ACKNOWLEDGEMENT

This work is partially founded by the EU (FP7 CONVERGENCE project) and by Telecom Italia.

REFERENCES

[1] D. Cheriton, M. Gritter, “TRIAD: a scalable deployable NAT-based internet architecture”, Technical Report (2000)”

[2] T. Koponen, M. Chawla, B.G. Chun, et al.: “A data-oriented (and beyond) network architecture”, ACM SIGCOMM 2007

[3] V. Jacobson, D. K. Smetters, J. D. Thornton et al., ”Networking named content”, ACM CoNEXT 2009

[4] A. Detti, N. Blefari-Melazzi, S. Salsano, M. Pomposini, “CONET: A Content Centric Inter-Networking Architecture“, ACM SIGCOMM ICN Workshop 2011

[5] PURSUIT project website: www.fp7-pursuit.eu

[6] 4WARD project website: www.4ward-project.eu

[7] CONVERGENCE website: www.ict-convergence.eu

[8] CCNx project web site: www.ccnx.org

[9] R. Pantos, “HTTP Live Streaming” Internet draft, http://tools.ietf.org/html/draft-pantos-http-live-streaming-01

[10] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, A. Markopoulou, “Microcast: Cooperative Video Streaming on Smartphones”, ACM MobiSys 2012

[11] J. Mol, J. Pouwelse, M. Meulpolder, D. Epema, and H. Sips, “Give-to-get: An algorithm for P2P video-on-demand," ACM MMCN '08

[12] http://netgroup.uniroma2.it/Andrea_Detti/ICNvideo/