Computer Networks 81 (2015) 272–288

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Mobile peer-to-peer video streaming over information-centric networks

Andrea Detti, Bruno Ricci, Nicola Blefari-Melazzi *

CNIT, University of Rome "Tor Vergata", Department of Electronic Engineering, Via del Politecnico 1, 00133 Rome, Italy

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form 4 February 2015
Accepted 22 February 2015
Available online 26 February 2015

Keywords:
Information centric networking
CCN
Video streaming
MPEG-DASH
Peer-to-peer
Mobile

ABSTRACT

Information Centric Networking (ICN) is a network paradigm alternative to the classic host-centric communication model: it provides users with content exposed as names, instead of providing communication channels between hosts. In this paper, we present a peer-to-peer application for live streaming of video content encoded at multiple bit rates. The application enables a small set of neighbouring cellular/Wi-Fi devices to increase the quality of video playback by using the Wi-Fi network to share the portion of the live stream downloaded by each peer via the cellular network. The application exploits the main functionalities of ICN: routing by name, in network caching and multicast delivery. Our work includes the implementation of a Java prototype of the application on a test-bed composed by Linux machines running the CCNx tool and streaming MPEG-DASH videos. We measured the performance of our solution and verified on the field that ICN simplifies the development of applications, as it provides built-in functions, which would be much more difficult to implement by relying on classical TCP/IP tools only.

1. Introduction

The classical Internet model relies on the IP *host-centric* paradigm, in which the network layer is used to transfer bits among hosts. This model is a very good fit for a set of current Internet applications, such as conversational (VoIP) or remote control (SSH) services, in which two specific hosts need to exchange data. However, most Internet applications nowadays use the network as a repository of contents, identified by names. And when these contents are requested by a very large number of users, their delivery on top of a host-centric IP network has required to introduce many incompatible, proprietary, content-oriented functionality, like name-based routing, caching, multicasting, data replication. For instance, Content Delivery Networks heavily exploit such functionality.

The research community is proposing a new network paradigm, called Information Centric Networking (ICN), with the aim of harmonizing, simplifying and making more efficient the handling of content within the network [1]. ICN proposes an evolution of Internet core functionality to inherently support content-oriented services in any kind of network: wide and local area networks, mobile ad-hoc or mesh networks. ICN rethinks network services and distributes information (or contents) identified by names rather than setting up bit pipes between hosts identified by addresses. When a user expresses an interest for a content to the ICN Application Programming Interface (API), the underlying ICN functionality takes care of routing-by-name the content request towards the "closest" copy of the content with such a name (e.g. original or replica server or an in-network cache), and of delivering the content back to the requesting user. Different ICN architectures have been proposed so far [2]; however, most past works (and this paper) take as reference the Content Centric Network (CCN) architecture [3], which is also

* Corresponding author.
E-mail addresses: andrea.detti@uniroma2.it (A. Detti), bruno.ricci@uniroma2.it (B. Ricci), blefari@uniroma2.it (N. Blefari-Melazzi).

http://dx.doi.org/10.1016/j.comnet.2015.02.018
1389-1286/

supported by a real implementation for Linux, MAC OS and Android devices, named CCNx [4]. A conceptually similar proposal to CCN is NDN, which is actively working to achieve analogous aims and has its roots in CCN [5].

Video streaming is one of the applications that motivated ICN and is expected to be one of the major sources of traffic for both fixed and mobile networks [6]. The video streaming community is rapidly adopting pull-based, adaptive schemes (e.g. MPEG-DASH [7]), which perfectly fit the ICN service model. Indeed, the HTTP GET primitive used to pull video segments can be easily replaced by a similar ICN GET primitive. Pull-based streaming schemes are used both for client-server and peer-to-peer streaming (PPS) applications [8].

In this paper, we present an ICN peer-to-peer application for live streaming of videos encoded at multiple bit rates (adaptive live video streaming). Peers are assumed to be a *small* set of neighbouring mobile cellular devices that cooperatively download a live video stream from the cellular interface and share downloaded video segments through a proximity channel (e.g. Wi-Fi Direct). The cooperation logic is designed to improve the playback quality perceived by a peer, with respect to the quality that the same peer could achieve by downloading the stream only by itself. The application exploits the CCN architecture [3]; as for video it uses the MPEG-DASH (Dynamic Adaptive Streaming over HTTP) streaming standard [7]. Our source code is freely available [22].

Although our solution presents some improvements with respect to existing applications, our primary goal is not to propose a better performing application, but to show how to exploit an ICN API, namely the CCN API, to simplify the application development. Indeed, if we had used the plain TCP/IP API, we would have had the burden of implementing and orchestrating on top of it routing-by-name, caching and multicast functionalities, which instead are built-in in CCN. In addition, it is worth noting that while there are many papers dealing with core ICN challenges, e.g. caching, routing scalability [21], transport mechanisms, security, etc. [9], only few of them are concerned with practical experiences on application design [11–13,15].

Our solution and code [22] have been tested both in an emulated environment and in a real cellular environment with mobile phones served by HSDPA networks of different operators.

2. Related works

2.1. CCN overview

CCN addresses contents by using unique hierarchical names that follow a URI syntax, e.g. ccnx:/foo.eu/video1. Long contents are split into chunks, uniquely addressed by names that contain the content name and the chunk number, e.g. ccnx:/foo.eu/video1/#x for the *x*th chunk of content ccnx:/foo.eu/video1 (in what follows we will omit the scheme identifier ccnx:/). To fetch a chunk, a receiver sends out an Interest message which includes the chunk name; then the network sends back the data within a Data message. Interest and Data messages are sent and received through any network interface available on a node; these interfaces, in the CCN framework, are called *faces*.

Fig. 1 reports the main elements of a CCN node, namely the Forwarding Information Base (FIB), the Pending Interest Table (PIT) and the Content Store (aka the content cache). A CCN node uses a name-based FIB to route-by-name Interest messages using a prefix match logic. A FIB entry contains a name prefix (e.g. foo.com) and the identifier of the upstream faces on which the Interest message can be forwarded towards available sources (e.g. face 2 in case of Fig. 1). In case an Interest message matches more than one FIB entry, a forwarding strategy selects one or a set of them on which to relay the Interest. Similarly to IP forwarding, we assume that the CCN forwarding strategy selects the face that provides the *longest prefix match*.

While the FIB is used to forward Interest messages, the PIT is used to forward back Data messages. During the Interest forwarding process, a CCN node leaves reverse path information ⟨chunk name, downstream face list⟩ in the PIT, where the downstream face list contains the list of faces from which the node received the same Interest. For instance, in case of Fig. 1 the node has received two Interests for the content ccnx:/foo.eu/video1/#x from faces 0 and 1. Only the first received Interest is forwarded; the following Interests are not forwarded but their downstream face identifiers are added to the downstream list of the PIT. When an Interest reaches a node having the requested chunk, the node sends back the chunk within a Data message. A node can have the chunk either because it is temporarily available in its Content Store or because there is a local repository application connected through a local face that permanently stores the chunks of a given content. The Data message is routed on the downstream path by consuming the information previously left in the PITs. The Data message is relayed hop-by-hop on all the downstream faces, so inherently providing in-network multicast distribution. Traversed CCN nodes temporarily cache forwarded Data messages in their Content Store so inherently providing in-network caching functionality.

To download a content, a receiver fetches all the related chunks by sending out a sequence of Interest messages. For flow control purposes, the receiver uses a receiver-driven approach [10] which consists in limiting the number of in-flight Interests through a congestion window (cwnd). The congestion window size may be constant or e.g. regulated by an Additive Increase Multiplicative Decrease (AIMD) congestion control mechanism.

CCNx [4] is a Linux-based implementation of CCN, whose faces are UDP or TCP tunnels. The software is mainly composed of: *ccnd* that implements the node functionality of Fig. 1 by using C code; a set of applications/libraries of which the most used are *ccnr*, which is a permanent repository of contents, *ccngetfile* and *ccnputfile* used to pull a content from or to push a content in a repository respectively, and *ccndc*, which controls the CCN FIB.

2.2. MPEG DASH

MPEG-DASH (Dynamic Adaptive Streaming over HTTP) [7] is the first standard for adaptive video streaming. In