OpenSatRelaying: a Hybrid Approach to Real-Time Audio-Video Distribution over the Internet

Franco Tommasi, Catiuscia Melle, and Valerio De Luca
Department of Engineering for Innovation, University of Salento, Lecce, 73100, Italy
Email: franco.tommasi@unisalento.it; catiuscia.melle@unisalento.it; valerio.deluca@unisalento.it

Abstract—In spite of massive research efforts devoted to the advance of the technologies for large-scale live distribution of audio-video IP streams, no totally satisfying solutions seem to have emerged so far. CDNs are still expensive and P2P-TV systems face substantial delay limitations. As the deployment of a global terrestrial IP multicast infrastructure still looks far, turning the attention to satellite-based multicast would seem a sensible choice. However, the cost of such technology has been a stumbling block until now. That is where the CHARMS architecture comes in. It is designed to take advantage of the formidable properties of the satellite without requiring a generic user to install any sort of satellite receiver or dish. Its cornerstone is the recursive terrestrial relaying of satellite streams a number of properly equipped hosts are able to receive. The present paper relates about the results of the OpenSatRelaying project, aimed at the implementation and testing of the CHARMS architecture.

Index Terms—Satellite communication, multicast communication, large-scale system, real-time streaming, P2P streaming

I. Introduction

Despite recent years' enormous interest in the technologies for distributing live audio-video on a very large scale by the Internet, it is still very expensive to satisfy *at the same time* all the following requirements [1]:

- Very large scale (e. g. $10^5 \div 10^7$ users);
- Live transmission (latency < 15s);
- Good audio and video quality (minimum 640x480 pixels, 24 frame/s).

IP multicast would be an obvious solution to the problem but, for well-known reasons, a global terrestrial IP multicast infrastructure is nowhere to be seen [2]. While waiting for epochal ISP policies shifts, massive infrastructure improvements or technology breakthroughs, satellite multicast might be considered for a potential answer to the question. Costs and accessibility concerns have frustrated the use of satellites until now and might account for an apparent lack of appreciation for the immense power of such medium among the research community. However, really expensive satellites are the ones video-broadcasters mostly use (and most dishes are aimed at). Conversely, several satellites are available whose abundant bandwidth is sold at a very reasonable cost. On the basis of the past experience in our lab activities, the price of a full year of a 512 kbit/s multicast downlink on one of such satellites may be below 50k€. Unfortunately, such an accessible and powerful resource does not come without drawbacks. The most serious is almost nobody owns an appropriate receiver and an antenna pointed at the cheaper satellites. Nor it is likely that a significant share of the users would consider the purchase of such items.

These observations and the desire to find a way to exploit the extraordinary power of satellite for the purpose of large-scale live audio-video streaming led to the design of the "Cooperative Hybrid Architecture for Relaying Multimedia Satellite Streams" architecture (CHARMS) [3], [4], prompted by the needs of the "*Campus Satellitare del Salento*" (CSS) [5], a platform for large-scale synchronous distance-learning. The basic idea is: the satellite multicasts an IP live stream over a large geographical area and a number (the higher the better) of hosts equipped for reception relay the stream to other peers by the terrestrial Internet. Recursively, whoever receives the stream may act as a relayer for other hosts. This paper presents an updated description of CHARMS and relates about the first experimental results achieved by it.

II. Related Works

Content Delivery Networks (CDNs) are the first commercial success story stemming from the awareness of the impracticability of IP multicast [2]. While very effective in many respects, their costs are momentous and increasing with the scale of the targeted audience [6].

Quite on the opposite side is the use of peer-to-peer (P2P) overlays for the same purposes: scalability is optimal and costs are almost non-existent. On the other hand reliability is minimal, delays are nearly unpredictable and content availability depends on users' behaviours and preferences [7], [8]. A relatively new development is the deployment of hybrid CDN-P2P infrastructures [9] in order to try to get the best of both worlds.

Projects based on hybrid satellite-terrestrial networks do exist, like [10]-[12]. By feeding CDNs through

Manuscript received January 8, 2014; revised March 16, 2014.
This work has been partially supported by a grant from Programma Operativo Nazionale (PON) Ricerca e Competitività 2007-2013
PON 254/Ric. Potenziamento del "CENTRO RICERCHE PER LA SALUTE DELL'UOMO E DELL'AMBIENTE" Cod. PONa3_00334. CUP: F81D11000210007.
Corresponding author email: franco.tommasi@unisalento.it.
doi:10.12720/jcm.9.3.248-261

satellite the backhauling costs are significantly reduced but the cost of other critical components of a CDN remains the same.

The Zattoo system [13] also belongs to the picture: its P2P overlay distributes a real-time stream received via satellite, but leaves mostly unexploited the satellite multicast capability.

The CHARMS architecture tries to integrate the best features of all the existing systems. It favours the serving of entire streams (against their splitting in chunks in P2P systems) as in CDNs, to keep the latency low. But it follows the cooperative approach of P2P systems. Finally, it taps into the enormous potential of satellite multicast.

In CHARMS two kinds of nodes cooperate: *Relayers* (*R*), equipped for satellite reception and *Hosts* (*H*), connected only to the terrestrial Internet. Thanks to satellite, a live content is broadcasted synchronously to a (theoretically) unlimited number of *Rs* which, wherever they are, get synchronously the same content. They accept to relay the multicast stream in a unicast flow to some *H* nodes. Recursively, each *H* accepts to do the same (if possible) with the other members of the overlay. A third type of nodes (*Leaves* or *L*) accepts but do not forward the relaying.

Compared to rival solutions, CHARMS offers:

- An economical advantage (vs. CDNs) when cheap satellites are used, provided the scale of the distribution is large enough;
- A reduction of traffic in the Internet backbone (relaying within the same AS is preferred);
- A much reduced latency (vs. P2P).

Before describing in detail CHARMS working, it is useful to clarify what CHARMS is not:

- CHARMS is NOT "CDN + satellite", since there is no idea of recursion in such systems;
- CHARMS is NOT "Satellite Multicast", since its users are able to receive satellite streams without any sort of satellite receiving paraphernalia;
- CHARMS is NOT "P2P-TV", since the streams are relayed without being split in small chunks;
- CHARMS is NOT "P2P + Satellite" (e.g. Zattoo) (no recursion and no multiple satellite receivers there).

The above stated, CHARMS shares some points with all four approaches. Namely:

- Like [10], [11] or [12] CHARMS exploits the power of satellite multicast;
- CHARMS relies on the willingness of all users to share the received streams;
- Only some hosts (CHARMS primary relayers) actually receive the original stream from satellite.

III. CHARMS Overlay Topology

In P2P streaming system, peer nodes form the so-called overlay network at the application layer on the top of the physical network. Most of the existing overlay topologies can be classified into multi-tree-based and mesh-based [14]. In multi-tree-based approaches, a stream is divided into several substreams and each substream is propagated through a different tree. In mesh-based approaches, peers split a stream into temporal chunks and dynamically establish several connections to exchange chunks within a temporal window with multiple neighbours simultaneously.

Simulation results in [14] show that mesh-based approaches achieve a better bandwidth utilization especially in presence of peers with heterogeneous connections. Furthermore, mesh-based systems are more robust in presence of the unpredictable changes of available bandwidth and against peer churning (i.e. peers randomly joining and leaving the overlay): this is because the propagation path of each chunk changes dynamically, whereas the propagation of a substream statically relies on a particular tree.

Nevertheless, unlike in tree-based overlays, the playback lag (i.e. the time from content generation at the media server and content reproduction on a peer) in mesh-based P2P streaming is highly variable [15] due to the dynamic nature of peer links. No current system seems to be able to bring the delay below the 1 minute limit. PPLive [16], one of the most popular P2P live streaming systems, has a maximum lag of about 150 seconds [15]. In another popular system, namely PPStream [17], playback delay during a large-scale broadcast event [18] varies from 120 to 210 seconds, depending on each peer location. Besides, experimental tests in [18] reveal also a lag of 80 seconds between two campus in the same LAN. In interactive real-time scenarios, where the user should have the opportunity to actively participate, delay should observe even more strict requirements: for example, during the real-time streaming of a lesson, a listener should have the chance to ask some questions in time. Another drawback of mesh-based overlay is represented by playback continuity impairments especially in the first minutes of a live transmission: the freezing ratio generally decreases with the increase of the peers in the system.

Reference [19] theoretically derives a minimum delay bound for chunk-based streaming systems and proposes a dynamic snowball streaming algorithm (DSB) that should minimize delay. DSB tries to heuristically adapt chunk upload scheduling to bandwidth and delay variations. However, it is mainly based on an oldest-chunk-first scheduling, which may increase the risk of content bottlenecks.

Agiler [20] reduces by more than 20% the average playback lag of traditional systems by clustering peers according to their Autonomous System (AS) and considers the network distance between the clusters and the media server to improve network efficiency.

Unlike many other systems, CHARMS builds dynamic relaying trees for the propagation of entire streams to drastically reduce the delay affecting most of the existing P2P streaming systems: as we have stated before, low delay is a key factor in interactive distance-learning scenarios, where real-time constraints are more strict than in other live events (such as sport events). Each H/L node is attached to two or more trees for the same stream, with trees rooted at different *R*. Each *R* node is equipped with antenna for satellite reception and with a broadband connection allowing the relaying of received streams toward a large number of *H*/*L* nodes. *R*, *H* and *L* nodes are coupled according to several matching logics that try to achieve a good tradeoff between nodes' QoS, network efficiency and computational efficiency. For these purposes, matching criteria consider not only the upload bandwidth and the workload of a relaying node, but aim also at favoring intra-AS and intra-LAN traffic. We will describe matching logics in detail in section V-B.

CHARMS overcomes also the issues related to peer churning and bandwidth fluctuations that affect traditional tree-based approaches: when a *H* or *L* node detects congestion issues on its receiving link or the departure of its relaying node, it can immediately swap to a secondary relayer, thus alleviating QoS degradation, video freezing and bandwidth throttling.

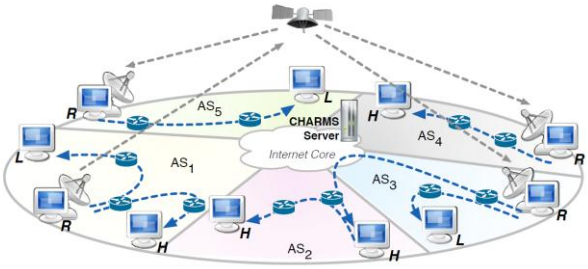

Fig. 1. Architecture overview

IV. ARCHITECTURE DESCRIPTION

CHARMS architecture (Fig. 1) is based on the cooperation among the following entities:

- The participating nodes, running a CHARMS client application, namely *CClient*, whether receiving from satellite or not;
- One or more CHARMS servers responsible for the overlay management and for contents availability tracking;
- A satellite configured for multicast transmission;
- some live content sources, which stream via satellite the captured events;
- A database storing real-time information about the overlay;
- Other services specific elements to support clients interactions with the system (*STUN servers*, *Web Server*, etc.).

While *R* nodes turn a multicast stream in more unicast ones, *H* nodes receive a unicast stream and relay it to one or more peers. Future upgrades will also implement a unicast to multicast relaying. Such relaying would be very effective where IP multicast is natively supported (LANs, CANs, MANs, one or more multicast peering Autonomous Systems).

The supervisor server (*S*) is needed to keep track of which nodes are receiving a stream and which among them are available to relay it. Its main purpose is to help newcomers to locate the most convenient relayer to contact, by matching requests from *H* or *L* nodes with offers from other *R* or *H* nodes. In addition it: a) monitors the overlay, coping with overlay variations (churning or unexpected crashes), b) assists nodes in getting in touch, c) ensures a minimum level of service to each node. *S* gathers client information into a dedicated database. Though *S* is unique in the tests, its functionalities are designed to be easily replicated and distributed.

The number of live events (*channels*) in CHARMS depends on the available satellite bandwidth. While *R* nodes can receive at the same time all the streams broadcasted by the satellite network, the terrestrial nodes (*H* and *L*) are allowed to receive a single event at a time.

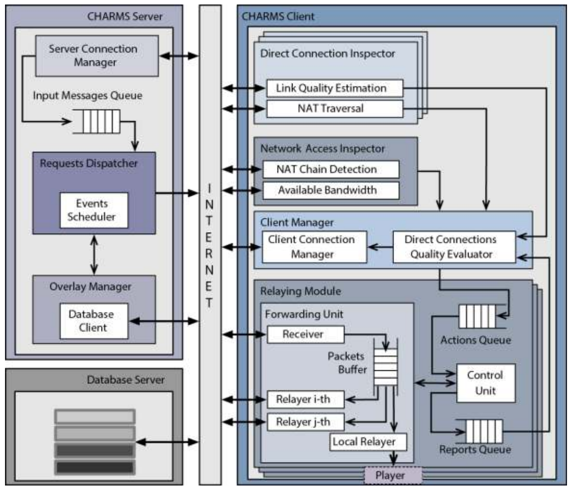

Fig. 2. High-level software architecture.

A. Core Modules

Fig. 2 represents the high level architecture of the system and the core modules of CHARMS Client (*CClient*, running on each *R*, *H*, *L*) and CHARMS Server (*CServer*, running on *S*) applications.

1) CHARMS server: CServer's main modules are:

- Server Connection Manager (SCM): manages the permanent SSL/TCP connections with the overlay nodes. It receives and validates the CHARMS messages, positioning them in the Input Messages Queue;
- Request Dispatcher (RD): pulls messages from this queue and, concurrently, processes and sends back a response to each client or contacts potential relayers on behalf of the requesting nodes;
- Events Scheduler (ES): is responsible for the orderly processing (and possibly delaying) of pending requests, according to the available resources in the system;
- Overlay Manager (OM): enforces the logic underlying CHARMS operations, selecting the best relayer to serve each request, querying and updating (by the Database Client sub-module) the Data Base.

2) CHARMS client: Its main modules are: the Client Manager (CM), the Direct Connections Inspector (DCI), the Network Access Inspector (NAI) and the Relaying Module (RM). CM implements *CClient* main logic, coordinating the other modules services. Following is a list of the tasks performed by each:

- NAT Chain Detection: discovers the type of NAT the client is behind [21];
- Available Bandwidth (AB): checks client outgoing and ingoing bandwidth;
- NAT Traversal: supports the establishment of a peer session between NATted nodes;
- Link Quality Estimation (LQE): periodically evaluates the state of the link on the peer session in order to guard against congestion or sudden crashes at the remote end;
- Relaying Module: receives and relays the real-time streams; it was implemented on the basis of ALRM [22], a modular and high-level library we developed to hide RTP/RTCP details;
- Direct Connections Quality Evaluator (DCQE): makes estimates of the state of the ongoing relayings, based on QoS information contained in the RTCP signaling and in the LQE output, to guarantee a minimum level of service to the overlay;
- Client Connection Manager: responsible for the general signaling with *S* (reports DCQE outputs to it).

A special Relayer is the Local Relayer which passes the stream to a local player - Apple QuickTime (QTP) or VideoLan Client (VLC) - for immediate viewing.

V. Implementation Details

CHARMS applications were developed from scratch under the UNIX OS (C, SUSv4) [23]. Later a Windows porting was added, based on the MinGW/MSYS [24], [25] environment enhanced by our "Network Abstraction Layer", which solves portability issues.

Signaling in CHARMS is based on a dedicated protocol and a terse XML tree definition and encoding.

The main chains of events in CHARMS are illustrated by Fig. 3:

- *S* receives (and answers to) messages from all nodes in the overlay (requests, periodical QoS and state information, etc.);
- *R* nodes receive real-time satellite streams through satellite antennas at their sites and periodically send to *S* the perceived QoS level;
- Users at hosts *H* visit the transmission live schedule hosted by a Web Server and choose the desired channel: a protocol handler allows the web browser to launch the installed *CClient* application that forwards the request to *S*. *S* looks for candidate peers for *H* and chooses R_i, the best suited to forward the stream.

A. Setup Phase

A node can join the CHARMS overlay through a setup phase, whereby it performs operations like login and characterization of the network access, sending to *S* the results. The login process allows the identification of the nodes. Due to the lack of globally unique IP addresses, a unique CHARMS cookie is attributed to each node at its first login and advertised subsequently. This step will help the adoption of an AAA [26] schema and a security policy, based on the dynamic assignment of Session Keys (K_s) for data and signaling encryption among peers.

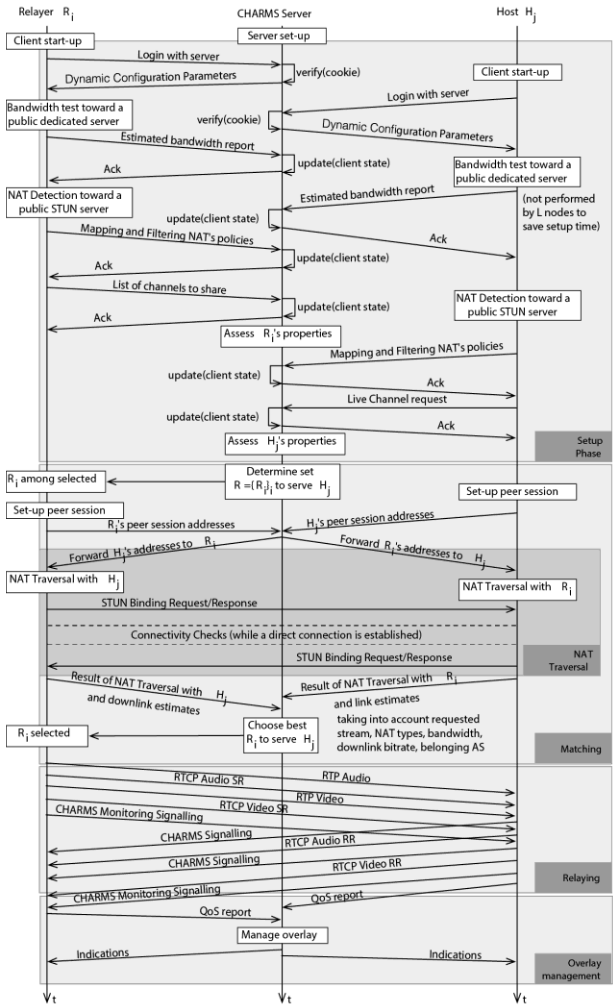

Fig. 3. Timeline of interactions among CHARMS nodes.

The sharing nodes (*R*, *H*, not *L*) use the AB Module for characterizing their network connection. The aim of this test is to detect an upper-bound for the available upload/download bandwidth based on the known connection types in a specific geographic area: such information is very important in the evaluation of the potential to meet the QoS requirements for stream relaying. It is not always an easy task, since several factors (peers' mutual positions, network distance, peers' workload, network congestion) may influence the estimate. However, further bandwidth evaluations will be based on service reliability provided by the node and on RTT monitoring during data exchanges with the CHARMS server.

The AB module implements a Packet Train/Pair Dispersion algorithm [27] to estimate the capacity of a node toward a dedicated server. The location and bandwidth capacity of that server must be carefully chosen so that the estimated value can be relied upon. Its address is provided by S during setup. The bandwidth test consists in sending trains of end-to-end packets of the same size. The dispersion is computed as the temporal distance between the last bit of the first packet and the last bit of the last packet. If N is the train length, L is the packet size and D the measured dispersion, the link capacity C is computed as:

$$C = \frac{(N-1)L}{D} \tag{1}$$

After the bandwidth estimate, a node executes the NAT Detection to discover, through a STUN server [21], whether they are behind a NAT and, in such case, the box's mapping and filtering policies.

Finally, each client specifies its role as R, H or L node and advertises its workload and capabilities. At the end of the setup phase, S knows everything it needs about a client, including the AS it belongs to.

B. Primary Relayer Selection

The key factor in the overlay construction is the optimal selection of a relayer node for each host node requesting a stream. A channel request from an H_j or L_k node is a trigger for S: it must figure out which R_i or H_m node is the best suited to become the R_{best} for the applicant. R_{best} is the node that, beside content availability, has the better end-to-end connectivity to the requesting node. As we will describe later, after this choice S selects one or more secondary relayers, called “fallback nodes”' (R_f), intended to replace R_{best} when it leaves the overlay or when the host experiences a bad QoS in stream reception.

The selection of R_{best} starts with a preliminary ranking of candidate nodes. The goals of this phase are two:

- The load balancing in the relaying tree, that is a fair distribution of host nodes requesting some streams among relayer nodes offering those streams; this could generally guarantee a higher overlay stability;
- The selection of a subset of nodes that with high probability would offer the best QoS performances; only these selected nodes will be involved in end-to-end connectivity test with the requesting node.

To fulfill both these requirements a list of candidate nodes is drawn up. Nodes that have not enough free outgoing bandwidth to relay the stream and those that are not NAT compatible with the requesting node are immediately discarded. The remaining nodes are ranked on the basis of the following criteria (sorted in decreasing order of importance):

- Nodes that would end up using all their free bandwidth, should their fallback sessions become active, are only considered as the last option;
- Nodes reporting reception problems are put on the bottom of the list, according to the extent of the problems;
- The choice of R_{best} tries to minimize the depth of the nodes in the overlay tree, in order to reduce the reception delay; besides, node churning causes more damages with longer chains;
- Nodes belonging to the same AS of the requesting node have higher priority, since intra-AS routing is usually faster;
- Nodes with longer recent sessions are preferred, because they could generally guarantee a higher overlay stability;
- Nodes that are serving a lower number of hosts are preferred for load balancing reasons.

For each candidate node we consider criterion 3 more important than criterion 6, because the relaying process does not suffer of excessive delay or packet losses until the saturation of the bandwidth capacity. Peer lifetime (5) can be an important factor for the system performances: as pointed out in [28], lifespan-based protocols can reduce disconnections ratio and the related costs by over 42%.

The best-ranked candidates are requested to start a point-to-point test with the requesting node. The test consists in a Request/Reply packets exchange just as the classic ping algorithm; however it uses the UDP transport level instead of ICMP. At this point a second selection phase starts: only nodes that reply within a short timeout (typically of 2 seconds) are considered for the final choice of the R_{best} node. They are ranked again according to the first two criteria of the previous phase and on the basis of:

- Loss rate, delay and jitter retrieved from test responses;
- The difference between their current free bandwidth and the average bandwidth needed for the stream relaying.

Node lifetime, loss rate, delay and jitter are classified into a set of confidence intervals whose number, ranging from a minimum of 2 to a maximum of 9, is proportional to the number of relayers in the system. The first node of the new ordered list is chosen as R_{best}. Then a NAT traversal procedure is started to establish a direct UDP connection between R_{best} and the requesting node. If the procedure fails, S chooses another R_{best} among the best-ranked candidates. If NAT traversal fails again, S schedules the selection of another R_{best} to be executed later: in this case, the procedure will start from a new ranking of candidate nodes.

C. Fallback Relayer Selection

The setup of a relaying session between a requesting node H/L and its R_{best} is time expensive due to the mandatory NAT Traversal procedure (see V-E). To avoid holes in the received stream, after the successful establishment of the primary relaying process, S chooses some fallback nodes R_f as we said before. The selection of R_f nodes is a bit different from the choice of R_{best}. We define the root relayer for a host as the root node of the stream relaying tree which the host belongs to: it is the only node in the tree that is receiving the stream directly from the satellite. The ideal and optimal criterion for the selection of a fallback node R_f for a host node H would be to minimize the number of common nodes in the chains connecting R_f and H to their respective root relayers. However this solution would not be efficient and scalable for high-depth trees because a comparison among long relaying chains would produce too heavy a computational workload. For this reason we resorted to a sub-obtimal criterion based on two requirements. The first requirement for a node to be included in the list of candidate $(R_f)_i$s is on the number of hops to their root relayer in the relaying tree. That number must be smaller than the same number for the node to be served (i.e. $depth[(R_f)_i] < depth(H)$). A second requirement demands the node to be served and the candidate $(R_f)_i$ to have different root relayers. However this requirement is not strict (i.e. they may have the same root relayer if no other node satisfy it). The rationale for this choice is trying to minimize the probability that both the host and the fallback nodes become orphan when a node leaves the overlay. If there are more suitable candidates, we also apply the same ordering criteria described in section V-B.

D. Fallback Sessions

While R_{best} forwards the stream, R_f and H/L nodes activate the LQE module that monitors RTT and loss rate on the path between them and refreshes NAT bindings.

This design choice increases the robustness of the overlay mesh so that sudden crashes and servant node departures can be swiftly managed with minimal changes in the overlay and damages in the playing stream. As soon as S diagnoses a decline of the QoS at the requester, caused by a problem on the path to it or in the status of R_{best}, the switch to R_f will be activated on the H/L nodes, while the session with R_{best} can be closed or paused. For unrecoverable sessions or relayer departures, S will rearrange nodes in the overlay on the basis of their current reports.

E. P2P Channels

In CHARMS, the assumption that all nodes are behind a NAT box is made; therefore a NAT Traversal procedure is needed to let each pair of nodes directly communicates. We developed our implementation of the UDP Hole Punching (UHP) technique, based on the use of the STUN protocol for performing connectivity tests and discovering the proper addresses of the involved nodes (or peer reflexive addresses), ending up in a procedure simpler than ICE [29].

Since UHP is expected to work fine with Endpoint Independent Mapping (EIM) policy NATs, S evaluates, for every pair of nodes, their reciprocal NAT compatibility (based on the results of the NAT detection test) to prevent that a node behind an Endpoint Dependent Mapping and Filtering NAT (EDM+EDF) will be paired with a node behind a NAT other than those implementing an Endpoint Independent Mapping and Filtering (EIM+EIF). S, acting as a rendez-vous server, exchanges the private and public address of each peer for every peer session, made up of five UDP channels (four for the stream and one for monitoring purposes). In each node, the NAT Traversal module, set in motion with these data, executes a STUN server and a STUN client to perform the connectivity tests for every of the five UDP channel, toward the public and private IP addresses. The STUN Binding Requests/Responses successful reception allow each node to discover the peer reflexive address of the other or, if no responses at all are obtained during a fixed timeout, its unreachability. At the end, each node communicates to S the result of the procedure.

F. Stream Relaying

Each considered multimedia stream incorporates RTP encoded audio and video sessions, requiring two RTP and two RTCP UDP channels. Due to NAT problems, no prior assumption on the UDP ports range for the relaying process is possible until NAT Traversal successful completion. Also, to guarantee the refresh of the NAT bindings for the unidirectional RTP channels (from R/H to H/L), CHARMS ack messages were implemented.

Each node forwards the received streams from its predecessor to the next destination as they are. Also the RTCP SR packets are forwarded unchanged from the original source: each node replies with a RTCP RR packet encoding values (such as loss rate) corresponding to the overall relaying chain, including both the satellite link and the terrestrial ones. In case of QoS deterioration, a receiving node cannot use these global estimates to know which of the links along the chain caused the problem, i.e. whether the loss comes from the last relaying or from a prior one. Here is where the DCQE module comes in: it tries to detect where the problem has occurred by combining the results of the LQE module and the RTCP estimates. There are three possibilities:

- LQE and RTCP indicate no problems: the perceived QoS level is good;
- The RTCP estimates signal a degradation, but LQE report no link congestion: then the problem must be behind the servant;
- LQE and RTCP together report a degradation of the peer session: the switch to R_f is performed.

In case (2), this logic, which is active in all nodes, will produce a switch as soon as one of them detects a problem with its direct servant: if no improvement is observed after a fixed timeout, then a switch to R_f is made.

When a switch is performed R_f quickly starts relaying the stream while the flow from R_{best} is blocked. Also, the LQE module activated with R_f produces an alert if the estimates go below fixed thresholds, so that another R'_f can be chosen.

S collects the LQE reports to prompt for new interconnections among the nodes or to rearrange the overlay after churning events. The introduction of fallback nodes distinctly improves the system reliability.

VI. Tests Results

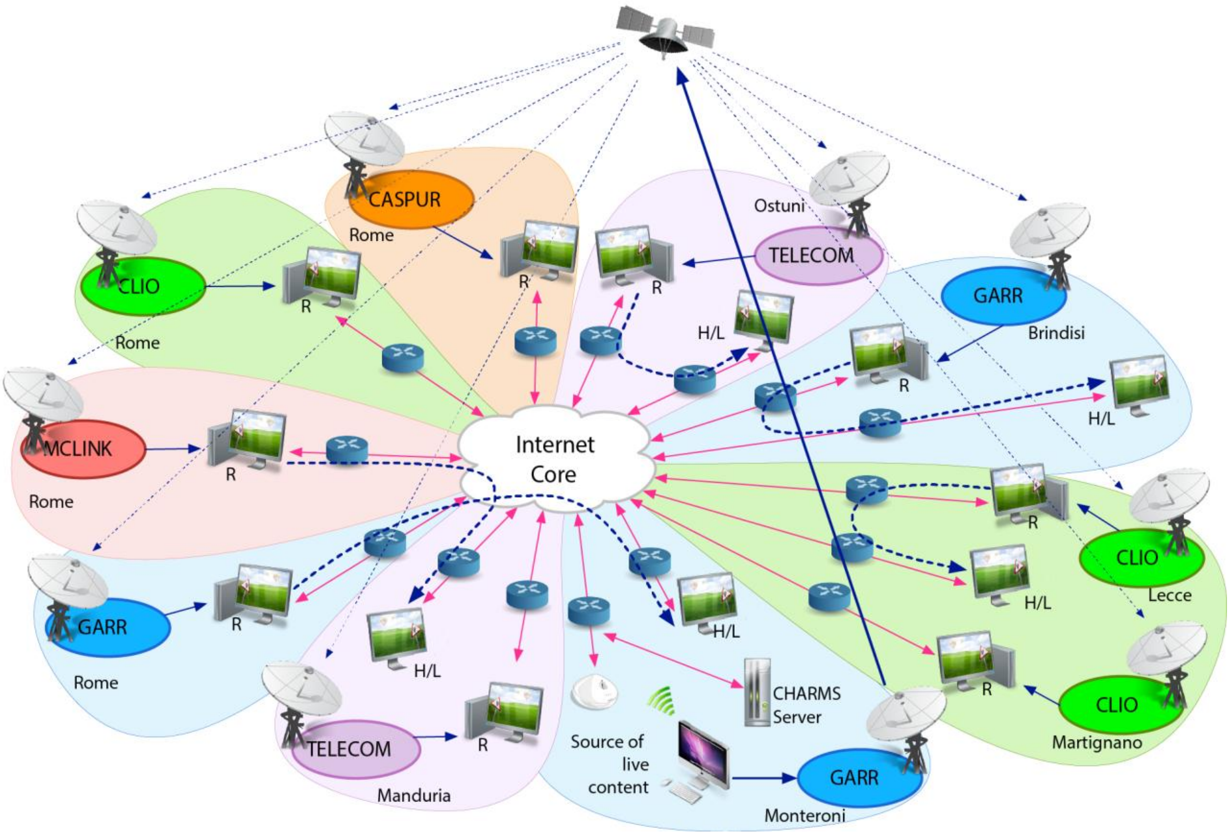

Fig. 4. Test bed scenario.

The tests of the architecture were conducted in the test bed of Fig. 4. While not the large-scale scenario CHARMS is aimed at, nonetheless it proved adequate to test the basic mechanisms and to gain insights about how well they could scale to large numbers.

The results focus on the following:

- Bootstrap delay, channel waiting, channel switching and fallback switching;
- Comparison between satellite and terrestrial reception;
- Signaling overhead;
- Video quality perceived on nodes;
- Delay and jitter.

A. *Test bed Setup*

Test components and operations:

- Audio-video source: an iMac (model 9.1, 2 GB RAM, MacOS 10.6.8). Acquisition (by incorporated microphone/camera) of an actual lecture;
- Encoding and streaming: same machine by *Apple QuickTime Broadcaster*, video spatial resolution 640x480 pixels, H.264/AVC [30] video codec, frame-rate 30 fps, target bitrate 512 kbit/s and audio MPEG4/AAC, 8000kHz (delivered to IP:224.100.0.40 over 4 distinct UDP channels);
- Transmission to satellite by a ViaSat® *LinkStar*$_{S2A}$™ router;
- A 1Mbit/s satellite (Eurobird 3™) uplink, IP multicast;
- Reception through *LinkStar* or iPricot® *IPR-SC DVB receiver-only* at satellite-equipped sites;
- Setup of a VPN infrastructure among the CSS sites for remote management.

CHARMS nodes visualize the received streams by QuickTime/VLC players.

TABLE I: ESTIMATED DOWNLINK AND UPLINK BITRATES: DOWNLINK HAS BEEN MEASURED WITH A ~8.9MB FILE TRANSFER FROM A NODE IN THE GARR NETWORK; UPLINK HAS BEEN MEASURED WITH THE CHARMS AB MODULE TOWARD THE SAME NODE.

Provider	Site	Downlink	Uplink
GARR	Rome	72.8	79.06/93.11/87.75
CLIO	Lecce	3.6	1.22/2.52/1.80
CASPUR	Rome	14.6	8.18/17.99/12.37
MCLINK	Rome	2.1	1.7/2.3/2.0
TELECOM	Manduria	2.0	0.26/0.35/0.32
TELECOM	Ostuni	2.0	0.52/0.63/0.554
CLIO	Martignano	0.9	0.23/0.3/0.278

Table I reports some properties of the Internet connections of the involved nodes.

Table II collects NATs' policies of the involved nodes detected with CHARMS NAT Detection module.

Table III reports CHARMS NAT traversal performance data.

TABLE II: MAPPING AND FILTERING NATS' POLICIES

Provider	Site	Mapping	Filtering	Public IP
TELECOM	Manduria	EIM	EDF	No
CLIO	Lecce	EIM	EIF	Yes
CLIO	Rome	EIM	EIF	Yes
GARR	Monteroni	EIM	EIF	No
GARR	Brindisi	EIM	EDF	No
TELECOM	Ostuni	EIM	EDF	No
CLIO	Martignano	EDM	EDF	No
MCLINK	Rome	EIM	EIF	Yes
GARR	Rome	EIM	EIF	Yes
CASPUR	Rome	EDM	EDF	No

EIM := ENDPOINT INDEPENDENT MAPPING; EDM := ENDPOJNT DEPENDENT MAPPING; EIF := ENDPOINT INDEPENDENT FILTERING; EDF := ENDPOINT DEPENDENT FILTERING

TABLE III: NAT TRAVERSAL PERFORMANCES

R-H pair	NAT side	R-H/H-R STUN packets	Time (s)	Sent/Received (kbytes/s)
Rome (MCLINK) Martignano (CLIO)	H	15/25	1.23	0.8/1.2
Lecce (CLIO) Ostuni (Telecom)	H	15/25	1.27	0.8/1.2
Rome (CASPUR) Lecce (CLIO)	R	20/15	0.06	21.7/18.4
Monteroni (GARR) Rome (GARR)	R	25/25	0.08	20.9/21.7
Monteroni (GARR) Rome (CASPUR)	both	20/15	1.11	1.1/0.8
Monteroni (GARR) Rome (MCLINK)	R	25/25	0.08	19.0/19.7
Monteroni (GARR) Lecce (CLIO)	R	25/25	0.13	11.9/12.4
Monteroni (GARR) Manduria (Telecom)	both	20/15	1.32	1.0/0.7
Rome (GARR) Martignano (CLIO)	H	22/19	1.15	1.2/1.0
Rome (GARR) Manduria (Telecom)	H	25/25	1.57	1.0/1.0

TABLE IV: STARTUP DELAY OF SOME POPULAR IPTV APPLICATIONS.

Zattoo	SOPCast	Joost	PPLive
5-8 sec	1.5 mins	4-8 sec	20 sec

B. *Bootstrap Delay, Channel Waiting, Channel Switching and Fallback Switching*

Table IV summarizes the startup delays for some popular IPTV applications retrieved from [31], [32].

In CHARMS we distinguish the time a node spends to join the overlay (bootstrap delay) and the time it has to wait before starting to receive a requested stream (channel waiting time). In the bootstrap phase *R* and *H* nodes spend 1 second for the bandwidth test and an average time of 5 seconds for the NAT detection test (ranging from a minimum of 1 second to a maximum of 9 seconds), for a total bootstrap time of about 6-7 seconds; this phase is a bit faster for *L* nodes, which skip the bandwidth test.

The channel waiting time on a *H*/*L* node is measured as the time between the request of a content and the reception of the first content datagram. It is made up of the time spent by the server to select the best candidate nodes (typically 1 second), the time spent waiting for the results of the point-to-point connectivity tests between the candidate nodes and *H*/*L* (2 seconds) and the time spent for the NAT traversal procedure between the selected relayer and *H*/*L* (1-2 seconds). Thus a *H*/*L* node in the overlay has to wait for an overall time of 4-5 seconds before receiving a stream (obviously provided there is at least one candidate relayer in the system with the requirements needed to properly satisfy *H*/*L*'s stream request).

From the channel waiting time derives the channel switching delay, which refers to the time a user has to wait before resuming the playback after he has switched from a channel to another. As stated in [33], in Zattoo the channel switching delay is worse than the startup delay: it has an average value of 8.4 seconds and ranges from a minimum of 5 seconds to a maximum of 14 seconds. On the contrary, a CHARMS *H*/*L* node generally waits about 5 seconds during a channel switching, since, besides stopping the reception of the previously received stream, the operations are exactly the same as for the first stream request. However, when the new requested stream is already available from its current relayer or fallback node, the *H*/*L* node simply waits about 2 seconds to directly start receiving the new stream from it and stop the reception of the previous one.

The fallback switching delay is the time a node has to wait before resuming the reception of a stream after a relayer failure/departure or after it has detected a sensible QoS degradation. The fallback switching procedure consists in starting the reception of the same stream from the fallback node R_f and signaling the end of the reception to the old relayer *R* (if it has not leaved the overlay): it requires about 2 seconds.

C. *Relaying Process Performances*

The effectiveness of the CHARMS delivery method is evaluated by comparing the reception of a stream by a satellite equipped-site with that of sites receiving it by terrestrial relaying. For simplicity, the analysis reported in Fig. 5 only deals with the RTP video packets exchanged during the run. Remembering the relaying process preserves the packets' sequence number, the red line plots the arrival times of the original packets from satellite (time=0 for the first packet) at a given *R* against their sequence number. The green line shows the average of the arrival times of the relayed packets at a number of *Hs* fed by the same *R* (time=0 for the first packet at each *H*). The ECDF in the figure covers, for all packets of all *Hs* fed by the same *R*, the difference between their arrival time and that of the packet with the same sequence number at *R* (time=0 for all first packets). It shows more than 99.9% packets at the *Hs* falls within 0.18 s from the corresponding ones at *R*, and are easily handled by video-buffering.

Fig. 6 focuses on a *R* node that is relaying toward 5 *H* destinations a stream received via satellite multicast: the incoming bitrate of the RTP stream is compared to the total outgoing UDP traffic, which includes also the RTCP subsessions and the traffic caused by LQE link monitoring on the 5th UDP channel between *R* and the *H* nodes.

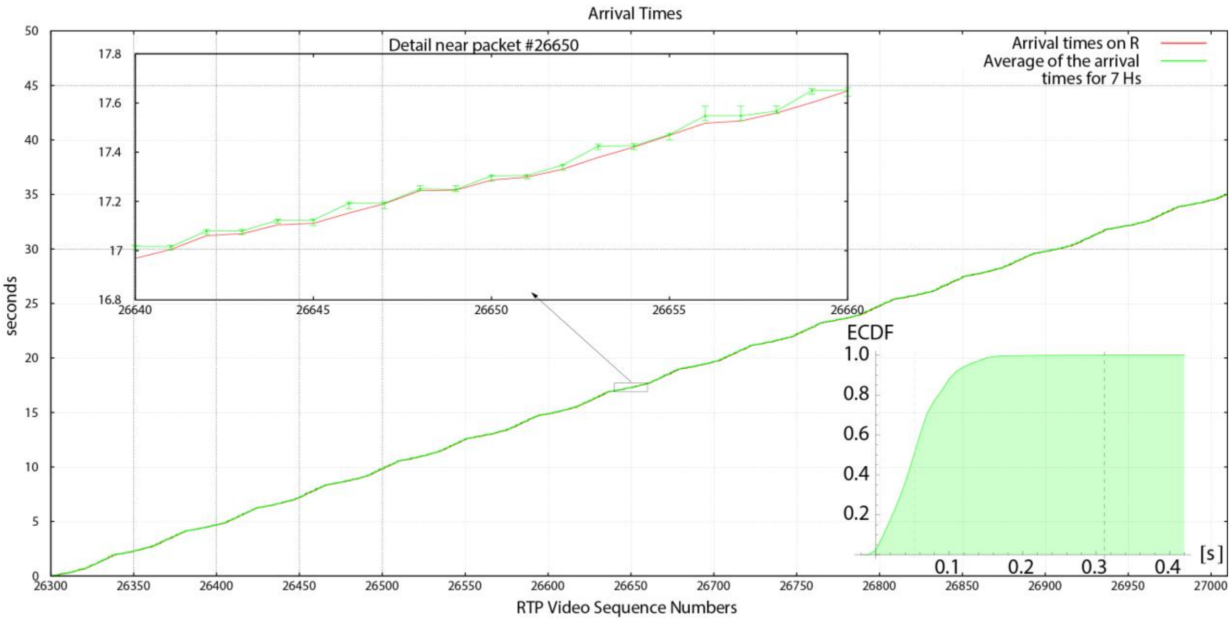

Fig. 5. Packets' arrival times from satellite compared with arrival times after relaying.

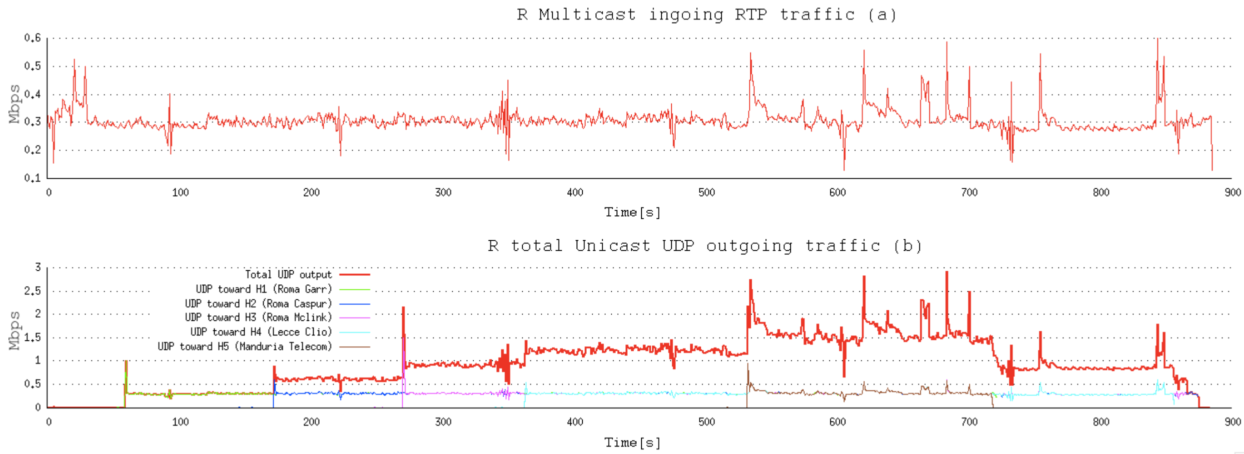

Fig. 6. Comparison on a *R* node between the incoming bitrate of a satellite multicast RTP stream (a) and the total outgoing UDP traffic toward 5 *H* destinations (b). The red line in (b) corresponds to the sums of the single sessions toward the different destinations. Peaks in the waveform are present in the initial phase of the relaying processes because of the larger traffic delivered by *R* to *Hs*, in order to provide them with enough packets to fill the players' buffers.

TABLE V: SIGNALING OVERHEAD WITH *S*

Node	Messages			Bytes		
	Toward S	From S	Total amount	Toward S	From S	Total amount
R_S	41	42	83	8062	11135	19197
H_S	24	24	48	4790	6000	11000

D. Signaling Overheads

1) With S: Table V points to the negligible SSL/TCP signaling overhead at *S*, measured by evaluating the input and output TCP bandwidth consumption for the establishment and maintenance of the overlay through the TCP CHARMS port. Results, collected on a ~700s run with 1 Relayer, 1 Fallback and 3 *Hs*, show an average SSL/TCP signaling of ~648 bit/s.

2) From relayers: Traffic from relayers to hosts is due to the stream relaying along with the RTCP SR and LQE signaling. Since signaling components account for $1*10^{-3}$ of the average bitrate, the Relayer outgoing bandwidth can be approximated as *B*N*, where *B* is the average bitrate for the A/V stream, *N* the number of peer sessions.

3) Fallback signaling overhead: The traffic between a node and its fallback (reported in Table) consists of the signaling involved by the LQE module over the five UDP channels, shifted of ~60s/5=12s for each channel to avoid peaks in the outgoing traffic.

4) Comparison with other systems: For a CHARMS node that is receiving a stream of 512 kbit/s and is relaying it to another node, the ratio between the signalling traffic and the overall traffic is 0.2% for the uplink and 0.1% for the downlink.

These bitrates are very low especially if compared to those of some popular P2P IPTV applications, as garnered by [34] and reported in Table VII.

As reported in [33], also Zattoo and Joost generate high overall overheads (56% and 13% respectively).

E. Video Quality Evaluation

We conducted some experimental tests to prove that the terrestrial Internet can replace to some extent the satellite network. Assuming that there are relayer nodes with enough bandwidth capacity and temporarily ignoring the satellite transmission, we focused our attention on the capability of domestic peers to receive a good quality stream. In particular, we sought to experimentally establish some resolution and bitrate settings that could allow a typical ADSL user to receive a video stream with good QoS and QoE levels.

TABLE VI: SIGNALING OVERHEAD BETWEEN PEERS.

The two RTCP subsessions with the *Relayer*	From *H* to the *Relayer*, the frequency of RTCP RR messages, correlated with the reception of the RTCP SR, results in a ~500 bit/s.
The CHARMS signaling on the two RTP subsessions with the *Relayer*	From *H* to the *Relayer*, a CHARMS Ack message (~200 bytes) is sent every 60s on the two UDP channels, causing ~54 bit/s.
One CHARMS session with the *Relayer*	On the 5th UDP channel a LQE signaling message of ~250 bytes is exchanged every ~60s, for a ~34 bit/s.
Five CHARMS sessions with the *Fallback*	LQE signaling with Fallback consists of ~300 bytes CHARMS message exchanged every ~60s, for a ~200 bit/s.
So, on a H node the total upload bandwidth consumption for CHARMS signaling is ~288 bit/s, plus the ~500 bit/s for the A/V RTCP signaling.	
On a R node instead, the RTCP SR signaling requires ~500 bit/s, plus the LQE signaling ~40 bit/s.	

TABLE VII: SIGNALING TRAFFIC RATIO OF SOME POPULAR IPTV APPLICATIONS.

Traffic (%)	PPLive	PPStream	SOPCast	TVAnts
Upload	2.2	10.8	13.6	7.8
Download	19.2	25.8	48.5	18.0

TABLE VIII: MOS AND DIV ESTIMATED ON *R* AND *H* AT DIFFERENT RESOLUTIONS AND BITRATES

Resolution	Bitrate (Mbit/s) on video source/R/H	MOS on R/H	DIV (%) on R/H
1920×1080	3.56/3.35/3.01	5.00/3.38	0.0/100.0
1280×720	1.83/1.72/1.71	5.00/4.99	0.0/4.0
720×576	1.04/0.95/0.95	5.00/5.00	0.0/0.0

TABLE IX: COMPRESSED VIDEO PARAMETERS

Codec	H.264
Frame rate	24 fps
Keyframe every	24
Number of frames	200
Duration	8 s

We encoded the same YUV source video (resolution 1920x1080 pixels, framerate 24 fps) with H.264/AVC at different resolutions and bitrates, reported in Table VIII.

Some other coding parameters, that are common to the three compressed videos, are reported in Table IX.

Then we delivered from a video source the encoded MP4 file as a RTP stream via terrestrial unicast to a CHARMS *R* node, which in turn relays it to a *H* node. In this second experimental phase, we considered the following new test bed:

- Video source: Linux PC behind GARR network in Rome;
- Relayer node (*R*): iMac (model 9.1, 2GB RAM, MacOS 10.6.8) behind GARR network in Monteroni;
- Host node (*H*): MacBook Pro (model 9.2, 8GB RAM, MacOS 10.8.3) in Telecom Italia ADSL network.

We reconstructed the video received on *R* and *H* and then evaluated some video quality parameters by means of EvalVid framework [35], [36]. We firstly computed PSNR [37] (Peak Signal-to-Noise Ratio), which compares the compressed frames with the original frames: it is defined as the ratio between the maximum pixel value in the original frame and the noise that affects the compressed frame.

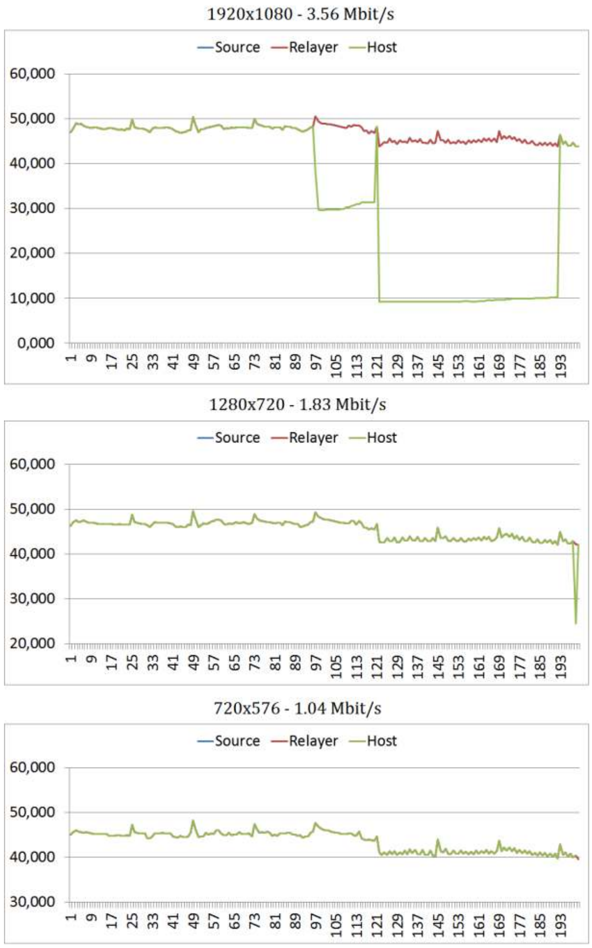

Fig. 7. PSNR on *R* and *H* compared to source PSNR in the three test cases.

In Fig. 7 PSNRs of video received by *R* and *H* are compared to source video PSNR for three different resolutions and bitrates. In the first scenario, characterized by the highest resolution and bitrate, the PSNR of the video received on *H* differs from both the source PSNR and the PSNR of the video received on *R* after the 96th frame, whereas in the other two scenarios the three PSNRs coincide. This agrees with the average bitrate discrepancy between the source video (3.56 Mbit/s) and the video received by *H* (3.01 Mbit/s), which is more prominent than the other two cases: we can conclude that the downlink capacity of *H*, which is behind a Telecom Italia ADSL, has been filled up in this case. The discrepancy can be pointed out in detail in the diagrams of Fig. 8: in the 1920x1080 scenario *H* receiving rate becomes substantially lower than *R* sending rate after 4 seconds (in both the cumulative and momentary forms); on the contrary in the other two cases the two rates keep about the same trend during the overall stream transmission.

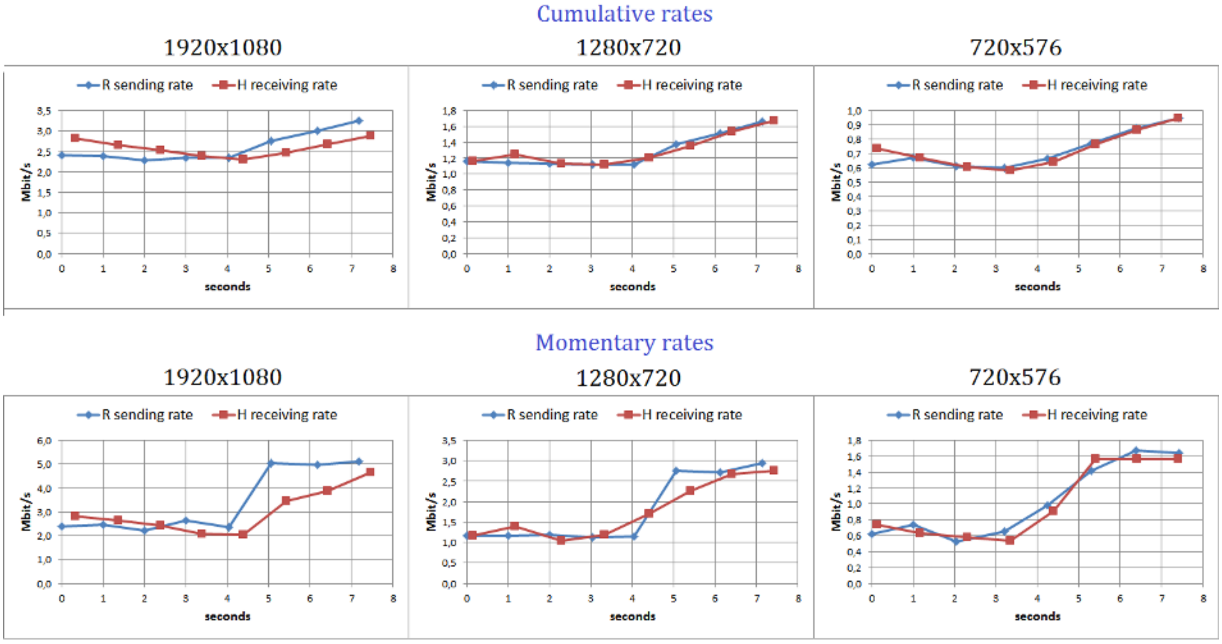

Fig. 8. Cumulative and momentary sending/receiving rates.

The third column in Table VIII collects the average MOS [37] (Mean Opinion Score) computed on *R* and *H* nodes for each resolution and bitrate: it represents the average video quality perceived by end users by means of an integer value between 1 and 5. In partial agreement with the PSNR degradation, we detected a lower MOS value for the 1920x1080 resolution and 3.56 Mbit/s bitrate. Sometimes however the MOS metric could fail to match the user perception: a user would not be satisfied if he notices few seconds of low quality even if the video has a good MOS value. For this reason, we evaluated also the DIV [37] (Distortion In Interval) metric, which restricts the video quality assessment within intervals of fixed size: we computed the percentage of frames having a low MOS within intervals of 25 frames.

In our analysis, even if the 3.38 MOS value is not too bad, we obtained a 100% DIV value, which highlights that in this case the real user satisfaction level would be even lower. In this case the average MOS summarizes into a unique value the overall stream reproduction, which consists of a former part (before the 96th frame) characterized by a good quality and a latter part (after the 96th frame) characterized by a bad quality. On the contrary the DIV value, which analyzes the video quality within narrow intervals, points out a very bad quality during part of the stream reproduction: it is a more severe metric, which however often reflects the real satisfaction level of a user, which is negatively influenced by very bad quality experienced within narrow intervals, even if the average quality during the whole stream reception is not too bad. We emphasize this negative result, no doubt related to bandwidth inadequacy in the considered domain, is found only in the case of a very high resolution.

The percentages of lost frames are reported in Table X for the three considered cases.

TABLE X: FRAME LOSSES FOR A RELAYER (R) IN A GARR NETWORK AND A HOST (H) IN A TELECOM ITALIA NETWORK

Resolution	I-frame losses (%) on R/H	P-frame losses (%) on R/H	B-frame losses (%) on R/H	Overall frame losses (%) on R/H
1920×1080	0.00/33.33	0.00/1.05	0.00/0.00	0.00/2.50
1280×720	0.00/0.00	0.00/0.52	0.00/0.00	0.00/0.50
720×576	0.00/0.00	0.00/0.52	0.00/0.00	0.00/0.50

TABLE XI: DELAY AND JITTER MEASURED ON A RELAYER (R) IN A GARR NETWORK AND A HOST (H) IN A TELECOM ITALIA NETWORK

Resolution	Delay (min/max) (s)		Jitter (min/max) (s)	
	R	H	R	H
1920×1080	0.011/0.160	0.051/0.417	0.0/0.148	-0.003/0.089
1280×720	0.073/0.199	0.111/0.278	0.0/0.085	0.0/0.127
720×576	0.012/0.179	0.056/0.345	0.0/0.127	-0.001/0.214

As explained in [37], a GOP (Group of Pictures) is made up of:

- Intra-coded frames (I-frames), which are independent from other frames;
- Predictively-coded frames (P-frames), which depend on previously coded frames;
- Bi-directionally predicted frames (B-frames), which depend on both previously and subsequently coded frames.

Finally, in Table XI we reported the minimum and maximum delay and jitter measured on *R* and *H* in the three test cases. We observed that there is no particular correlation with packet losses: even in the case we experienced a higher packet loss and QoE degradation we noticed low delay and jitter values.

F. Computational Workload

The CPU usage of a host node varies between 0.2% and 0.5% during a stream reception. The CPU usage of a relayer node varies between 1% and 4% during a relaying towards 5 host nodes. It tends to grow linearly with the number of served hosts. Clearly, the main constraint to system scalability is relayers' outgoing bandwidth and not their computational capacity. The simple propagation of entire streams through the overlay does not cause a too heavy computational workload.

The computational workload of the CHARMS server is very low, since it does not contribute to the stream propagation. It simply keeps track of the existing nodes in the overlay and matches requests from host nodes with offers coming from relayer nodes. The mere presence of a great quantity of nodes forming a sufficiently stable overlay generates no work at all for the server: this is also because each node, once it has been provided with a primary and secondary relayer, is able to autonomously monitor QoS and eventually switch to its fallback. The maximum amount of workload for the server occurs only in presence of flash crowds (i.e. several nodes joining the overlay within a short time interval) or in presence of several nodes leaving the overlay (in a short time period). Even in those critical cases, for the actual server machine used in trial (iMac Intel Core 2 Duo 2.66 Ghz, 2GB RAM, MacOS 10.6.8), the response time of the server is always below two seconds and the CPU usage never exceeds 4%. We should also point out that the server operations mainly consist in read or write accesses to the database: in our implementation we take advantage of PostgreSQL [38] built-in support for regular B-tree and hash indexes, which allow a very efficient query execution (and thus information retrieval) in logarithmic and constant time respectively; furthermore, PostgreSQL also manages the concurrent execution of queries, with fine-grain locks that can be applied to single database rows. These expedients contribute in keeping under control the server computational workload.

Even though we used a single server in our experimental tests, we can easily set up several replicas of the server that can replace it in case of failure. In this way, the single point-of-failure issues can be avoided too.

Since a Linux machine is able to efficiently manage a maximum number 1600000 sockets [39], we can suppose it is also the maximum number of clients that could be handled by a single CHARMS server. Nevertheless, we think an array of multiple CHARMS server would further improve the system scalability up to millions of client nodes.

G. System Scalability

To get some insights about the overall system scalability, let us consider the following scenario with N_R relayer nodes equipped with satellite antenna and a broadband terrestrial connection. Let us suppose each relayer node is able to efficiently relay a stream towards *Children*(R) or *Ch*(R) host nodes, whereas each host node is able perform further relaying towards *Children*(H) or *Ch*(H) host nodes of the underlying layer of the tree. If we denote with *depth* the depth of each relaying tree, the total number of served nodes in the system would be:

$$\begin{aligned} N &= N_R + N_R Ch(R)\sum_{i=0}^{depth}[Ch(H)]^i = \\ &= N_R\left[1+Ch(R)\left(\frac{Ch(H)^{depth+1}-1}{Ch(H)-1}\right)\right] \end{aligned} \tag{2}$$

It is worth noting that, unlike mesh-based overlays, CHARMS can provide with high probability a reliable upper bound to the playback lag on each receiving node: indeed, matching logics avoid the construction of too depth relaying trees; furthermore, if delay increases on one or more links of a relaying path (for example due to congestion events), nodes detecting QoS degradations properly switch to their fallback nodes, which can guarantee an acceptable delay. However, even though the experimental tests in VI-C and VI-E show good performance in GARR and CASPUR networks, nowadays commercial networks for domestic use impose some limitations. Clearly, the scalability of the system would be distinctly improved if there were some broadband nodes also among H_j host nodes. Even if H_j nodes can typically contribute only to a lesser extent to the propagation of entire streams, constant advances in bandwidth availability of domestic Internet accesses tend to progressively reduce the efficiency gap between our relaying method and chunks based relaying. In the meantime, Multiple Description Coding [40] techniques could allow a better use of bandwidth-constrained host nodes, even though the overhead impact on playback delay has to be evaluated too.

VII. Conclusions

While waiting for the resources needed to start a massive field trial, the experimental results show the described architecture has the potential to efficiently distribute good quality live content (delay < 8s after a 3-tier relaying) with good scalability (very low CPU load on the relayers and an even lower CPU usage on the CHARMS server). A crucial factor for the triggering of the multiplicative effect of the recursive relaying is the statistical distribution of the outgoing bandwidth available to the users' population. In regions where most domestic connections do not presently support a comfortable relaying of an entire stream (e.g. South-Italy) the expected snowball effect can hardly take place. However the same tests suggest that where and when even domestic connections allow the relaying of one of more streams the exponential growth of the relayers may act as anticipated, thus bringing about the awaited economical advantages over concurrent solutions. A substantial field for improvement would come from the splitting of a single stream in smaller portions with Multiple Description Coding [40] techniques, which would allow the smaller ADSL connections to participate in the establishment of the desired multiplicative effect. However, we should verify that the overhead introduced by such techniques would not affect playback delay significantly in an interactive and real-time scenario. Another field of improvement would come from the adoption of HTTP adaptive streaming instead of the more traditional technique used in the test. Both developments are currently under way.

ACKNOWLEDGMENT

The authors would like to thank Dr. Alfredo Grieco of the Politecnico di Bari for many useful suggestions and insights.

REFERENCES

[1] H. Ketmaneechairat, "A survey and comparison of some popular iptv applications," in *Proc. 8th International Conference on Computing Technology and Information Management*, vol. 1, 2012, pp. 58–63.

[2] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the ip multicast service and architecture," *Network, IEEE*, vol. 14, no. 1, pp. 78–88, 2000.

[3] F. Tommasi, S. Molendini, E. Scialpi, and C. Melle, "Charms: Cooperative hybrid architecture for relaying multicast satellite streams to sites without a satellite receiver," in *Proc. IEEE International Conference on Wireless Communications, Networking and Information Security*, 2010, pp. 269–276.

[4] F. Tommasi and C. Melle, "Large-scale terrestrial relaying of satellite broadcasted real-time multimedia streams," *International Journal of Computer Networks & Communications*, 2011.

[5] F. Tommasi, A. Campa, and S. Molendini, "The campus satellitare del salento: A scalable and transparent e-learning application," in *Proc. International Conference on Software in Telecommunications and Computer Networks, SoftCOM*, 2006, pp. 363–368.

[6] Z. Zhuang and C. Guo, "Optimizing CDN infrastructure for live streaming with constrained server chaining," in *Proc. IEEE 9th International Symposium on Parallel and Distributed Processing with Applications*, 2011, pp. 183–188.

[7] I. Ullah, G. Doyen, G. Bonnet, and D. Gaiti, "A survey and synthesis of user behavior measurements in P2P streaming systems," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 3, pp. 734–749, 2012.

[8] J. Mendes, P. Salvador, and A. Nogueira, "P2P-TV service and user characterization," in *Proc. IEEE 10th International Conference on Computer and Information Technology*, 2010, pp. 2612–2620.

[9] Z. Lu, X. Gao, S. Huang, and Y. Huang, "Scalable and reliable live streaming service through coordinating CDN and P2P," in *Proc. IEEE 17th International Conference on Parallel and Distributed Systems*, 2011, pp. 581–588.

[10] E. Feltrin, Y. Guinamand, and S. Agnelli, "The VERDI project: An interactive network integrating satellite and terrestrial technologies," in *Proc. International Conference on Information Technology: Research and Education,* 2003, pp. 365–369.

[11] A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "Virtual machine provisioning through satellite communications in federated cloud environments," *Future Gener. Comput. Syst.*, vol. 28, no. 1, pp. 85–93, Jan 2012.

[12] H. Hlavacs, M. Haddad, C. Lafouge, D. Kaplan, and J. Ribeiro, "The CODIS content delivery network," *Comput. Netw. ISDN Syst.*, vol. 48, no. 1, pp. 75–89, May 2005.

[13] H. Chang, S. Jamin, and W. Wang, "Live streaming performance of the Zattoo network," in *Proc. 9th ACM SIGCOMM Conference on Internet Measurement Conference*, New York, NY, USA: ACM, 2009, pp. 417–429.

[14] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live P2P streaming approaches," in *Proc. 26th IEEE International Conference on Computer Communications. IEEE*, 2007, pp. 1424–1432.

[15] X. Hei, Y. Liu, and K. Ross, "Inferring network-wide quality in P2P live streaming systems," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 9, pp. 1640–1654, 2007.

[16] [Online]. Available: http://www.pplive.com

[17] [Online]. Available: http://www.ppstream.com

[18] W. Liang, J. Bi, R. Wu, Z. Li, and C. Li, "On characterizing ppstream: Measurement and analysis of P2P IPTV under large-scale broadcasting," in *Proc. Global Telecommunications Conference, GLOBECOM 2009. IEEE*, 2009, pp. 1–6.

[19] Y. Liu, "Delay bounds of chunk-based peer-to-peer video streaming," *Networking, IEEE/ACM Transactions on*, vol. 18, no. 4, pp. 1195–1206, 2010.

[20] D. Huang, J. Zhao, and X. Wang, "Agiler: A P2P live streaming system with low playback lag," in *Proc. 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2010, pp. 1–10.

[21] D. Macdonald and B. Lowekamp, "NAT behavior discovery using session traversal utilities for NAT (STUN)," *RFC 5780 (Experimental), Internet Engineering Task Force*, May 2010.

[22] F. Tommasi, V. De Luca, and C. Melle, "A Library for RTP Relaying and QoS Monitoring in Application Layer Multicast," in *Proc. Fifth International Conference on Ubiquitous and Future Networks*, 2013, pp. 418–423.

[23] THE Open GROUP. [Online]. Available: http://pubs.opengroup.org/onlinepubs/9699919799/

[24] MinGW. Minimalist GNU for Windows. [Online]. Available: http://www.mingw.org

[25] MinGW. Minimalist GNU for Windows. [Online]. Available: http://www.mingw.org/wiki/msys

[26] L. chuan Gu, S. wen Li, Z. Cheng, and Y. hua Zhang, "A new network access control method based on diameter protocol," in *Proc. International Conference on Communications and Mobile Computing*, vol. 3, 2009, pp. 600–604.

[27] C. Dovrolis, P. Ramanathan, and D. Moore, "Packet-dispersion techniques and a capacity-estimation methodology," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 963–977, 2004.

[28] F. E. Bustamante and Y. Qiao, "Web content caching and distribution," in *Friendships That Last: Peer Lifespan and Its Role in P2P Protocols*, F. Douglis and B. D. Davison, Eds., Norwell, MA, USA: Kluwer Academic Publishers, 2004, pp. 233–246.

[29] J. Rosenberg, "Interactive connectivity establishment (ICE): A protocol for network address translator (NAT) traversal for offer/answer protocols," in *Internet Requests for Comment*, RFC Editor, Fremont, CA, USA, Tech. Rep. 5245, April 2010.

[30] H. 264: Advanced video coding for generic audiovisual services. Recommendation H.264 (04/13). ITU-T and ISO/IEC.

[31] M. Alhaisoni, A. Liotta, and M. Ghanbari, "Resource trade-off in P2P streaming," in *Proc. 17th International Packet Video Workshop*, 2009, pp. 1–8.

[32] [32] M. Alhaisoni and A. Liotta, "Characterization of signaling and traffic in joost," *Peer-to-Peer Networking and Applications*, vol. 2, no. 1, pp. 75–83, 2009.

[33] M. Alhaisoni, A. Liotta, and M. Ghanbari, “Performance analysis and evaluation of P2PTV streaming behavior,” in *IEEE Symposium on Computers and Communications,* 2009, pp. 89–94.

[34] T. Silverston, O. Fourmaux, A. Botta, A. Dainotti, *et al.*, “Traffic analysis of peer-to-peer iptv communities,” *Comput. Netw.*, vol. 53, no. 4, pp. 470–484, March 2009.

[35] J. Klaue, B. Rathke, and A. Wolisz, “EvalVid - A framework for video transmission and quality evaluation,” in *Proc. 13th Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, ser. Lecture Notes in Computer Science. Springer, 2003, pp. 255–272.

[36] Institut für Telekommunikationssysteme: Evalvid, [Online]. Available: http://www.tkn.tu-berlin.de/menue/research/evalvid/

[37] J. Gross, J. Klaue, H. Karl, and A. Wolisz, “Cross-layer optimization of ofdm transmission systems for mpeg-4 video streaming,” *Comput. Commun.*, vol. 27, no. 11, pp. 1044–1055, July 2004.

[38] Postgresql Documentation. [Online]. Available: http://www.postgresql.org/docs/

[39] A programmer’s site. [Online]. Available: http://shenfeng.me/how-far-epoll-can-push-concurrent-socket-connection.html

[40] V. Goyal, “Multiple description coding: Compression meets the network,” *Signal Processing Magazine, IEEE*, vol. 18, no. 5, pp. 74–93, 2001.

Franco Tommasi graduated from University of Pisa, Pisa, Italy, in Electronic Engineering in 1984. In the same year he joined CSELT, Turin, Italy, where he became involved in image processing. In 1988 he joined the Istituto per la Ricerca Scientifica e Tecnologica, Trento, Italy, where he made researches in image motion analysis. He has been head of software development at Cigraph, Venice, Italy. He is now associate professor at the Engineering Faculty of the University of Salento, Italy. His research interests presently include satellite internetworking, large-scale Internet audio-video distribution and the applications of satellite to distance learning.

Catiuscia Melle graduated magna cum laude from University of Salento, Lecce, Italy, in Information Engineering in 2009. Currently she is a Temporary Research Fellow at LIIS laboratory at University of Salento. Her present research interests include networking and content distribution applications.

Valerio De Luca graduated magna cum laude from University of Salento, Lecce, Italy, in Information Engineering in 2009. Currently he is a PhD student at LIIS laboratory at University of Salento. His present research interests include distributed systems for real-time audio- video streaming and video QoE.