CoreCast: Efficient Live Streaming in the Core-Edge Separated Internet*

Loránd Jakab
Universitat Politècnica de Catalunya
ljakab@ac.upc.edu

Albert Cabellos-Aparicio
Universitat Politècnica de Catalunya
acabello@ac.upc.edu

Jordi Domingo-Pascual
Universitat Politècnica de Catalunya
jordid@ac.upc.edu

ABSTRACT

The increasing bandwidth offered by home broadband connections makes delivering live streaming and IPTV content increasingly more attractive over the Internet. Solutions range from the least bandwidth-efficient unicast point-to-multipoint, through peer-to-peer, to the most efficient multicast delivery mechanisms. But IP multicast does not scale well to the global Internet and today's practical implementations are restricted to "walled garden" scenarios, where content is entirely delivered inside the ISP. We propose CoreCast, a new live streaming delivery mechanism that takes advantage of the likely introduction of the Locator/Identifier Separation Protocol (LISP). LISP is being pushed as a possible solution to the Internet's routing scalability problem. We show that with minor modifications to LISP we can have a scalable global multicast solution.

1. INTRODUCTION

In today's IPTV and live streaming landscape we see two major directions. On one hand we see the "walled garden" approach of ISPs to provide several classical TV channels to their customers in triple-play packages consisting of IPTV, IP telephony and Internet access. On the other hand we have collaborative transmission of channels or events by self-organizing peer-to-peer networks.

The first approach offers guaranteed quality of service but limits the user to a relatively small fixed channel list. Channels are received by classical satellite or terrestrial transmission, encoded for packet based transmission at the ISP head-end and transmitted to customers using IP multicast [1]. This approach has the main disadvantage of putting the ISP in control of choosing the list of available channels.

In recent years a lot of research effort was devoted to find an alternative that puts users in control of choosing between an almost unlimited number of channels, including some less popular ones, which wouldn't normally be included in a typical TV package offering. As a result, we have seen peer-to-peer based live streaming systems like Zattoo, Joost, PPLive, Sopcast, etc. become popular with Internet users. While these streaming systems scale well, there are several challenges they still have to overcome, such as decreasing channel changing delay, which is in the order of tens of seconds (measured as 10-20 s in [2]) and the playback lag, which can be several minutes.

We believe both approaches have their place, but see the need for a different approach, that would enable users to choose between a vast number of channels with high reliability and low delay. To that end, we propose *CoreCast*, an explicit multicast scheme which overcomes the scalability issues of IP multicast and offers a network-layer solution for live streaming, in contrast with the application-layer approach of P2P schemes. For each multimedia payload CoreCast will only send one copy, and inform responsible routers of all destinations. An important advantage of CoreCast is that it doesn't require all routers in the Internet to be modified as in the case of other multicast protocols. Instead it is based on a core-edge separation protocol, modifying only two routers in the path of a packet.

Core-edge separation protocols have been proposed to solve the routing scalability issues of the current Internet architecture. They would use separate addressing namespaces for transit networks and edge networks, to keep the routing tables of core routers manageable. Since the problem they try to solve is important to the Internet community, it's likely that one of the proposed solutions will be adopted in the future. As of now these proposals are still work in progress, open to improvements. This presents a window of opportunity to introduce native support for an efficient multimedia delivery system in the Internet such as CoreCast.

While our proposal is generic enough to be applicable to any core-edge separation protocol which uses a map-and-encap scheme, we will explain it in terms of the Locator/ID Separation Protocol (LISP) [3]. Our choice is motivated by the amount of support received by this protocol by both one of the biggest router manufacturers and academia.

2. LISP

In LISP terminology, the Internet core (transit networks) uses Routing LOCators (RLOCs) as namespace for the addressing, and edge networks use Endpoint IDentifiers (EIDs). To facilitate transition from the current Internet, both namespaces use the existing IP addressing scheme. An additional plane, the Mapping System facilitates the "glue" between the two spaces, by returning the RLOC(s) corresponding to an EID. The advantages of this approach and related literature can be found in [3].

The life of a packet in the LISP-enabled Internet is as follows: it travels within the autonomous system (AS) using currently deployed mechanisms until it reaches the border router. This router is

*This work has been partially supported by the Ministry of Innovation, Universities and Enterprise of the Generalitat of Catalonia under scholarship number 2006FI-00935 and the Spanish Ministry of Science and Technology under contract TSI 2005-07520-C03-02

SIGCOMM'09 August 17–21, 2009, Barcelona, Spain

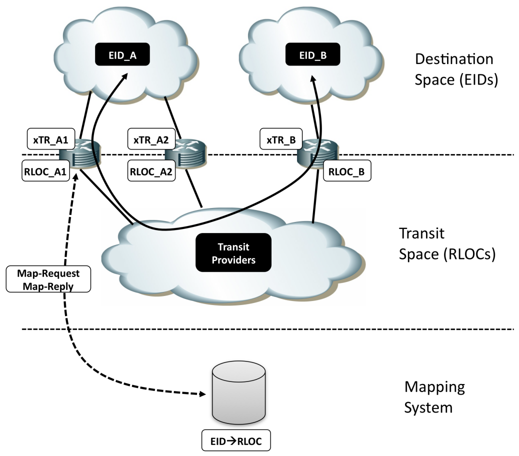

Figure 1: LISP Architecture

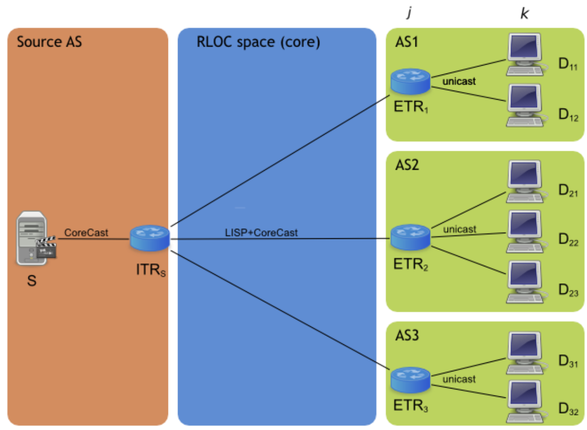

Figure 2: CoreCast

called the *ingress tunnel router* (ITR) in LISP terminology because it is the ingress point to the tunnel to the border router of the destination AS, the *egress tunnel router* (ETR). Since a border router can implement both functions, we will use the term Tunnel Router (xTR) for this kind of device. Consider Fig. 1 for example. A host with EID_A wants to send a packet to EID_B. It reaches xTR_{A1}, which takes the destination address (EID_B), looks it up in the mapping system, which returns $RLOC_B$. xTR_{A1} encapsulates the packet in a LISP header, sends it to xTR_B, which decapsulates it and then it gets delivered to the destination.

3. CORECAST

CoreCast is a simple, one-to-many multicast protocol with low deployment cost, incrementally deployable, exploiting features introduced by LISP in order to reduce redundant traffic. To implement CoreCast, a small number of modifications to the current LISP specification are required. This shouldn't be a problem, since LISP is still marked as work in progress.

Consider the following scenario: a broadcaster has a large number of clients (denoted by k) for live streaming content or an IPTV channel. These clients are dispersed over a number of j Autonomous Systems (ASes). When using unicast, the same content has to be sent k times by the source node S to reach all clients. Using CoreCast, the content is sent once to the ITR of the source node (ITR_S) along with a list of destinations, which in turn sends one copy to each involved ETR ($ETR_1 \ldots ETR_j$), and these ETRs send out one copy to each destination node inside of their respective ASes, see Fig. 2.

The most important parameter for the efficiency of CoreCast is the grouping coefficient γ. We define γ as the ratio between k and j. When γ is 1 (its lowest value) using CoreCast is actually detrimental, due to the overhead. As the value increases, the savings can become important. We expect γ to be very high because usually users of to the same domain are likely to be interested in the same important live events, such as sports transmissions.

CoreCast works as follows: a multimedia stream has a list of destinations D_i (where $i = 1 \ldots k$) and is divided into chunks of 1200 to 1400 bytes [2] c_m (where $m = 1 \ldots C$) by the streaming server application on S. Each chunk of the stream (c_m) is transmitted along with a special CoreCast header to an IP address with a special (local) meaning for ITR_S. When the ITR receives the packet, it processes the CoreCast header, which instructs the router to hash the content ($hash(c_m)$), cache it, but forward nothing. Please note that we use a special destination address because this way ITRs are not required to look inside the payload of *all* packets, the special destination address will trigger the processing of the next header *only* in the case of CoreCast packets. Since the ITRs and S belong to the same administrative domain, this is not an issue.

Then, for each client D_i, S transmits an IP packet towards the special address, also along with a CoreCast header. The header includes the hash of the chunk along with the IP address of a client. ITR_S processes this header and asks the LISP mapping system for the RLOC of the ETR corresponding to that destination address. It keeps tracks of all ETRs where c_m is sent, so for each ETR_j the chunk is only sent once, similar to the case of sending from S to ITR_S.

All these packets are forwarded in the Internet core as regular LISP encapsulated packets, except that they include a special bit in a reserved field of the LISP header. This bit instructs CoreCast-capable ETRs to process them according to the CoreCast protocol. When a CoreCast packet is received, the ETR checks the CoreCast header. If the packet contains content, it hashes the content and forwards it using unicast to the client. If the packet contains a hash, then it looks up the corresponding content in its cache, replaces the hash with the content and forwards it to the client using unicast.

As we have mentioned, CoreCast is incrementally deployable, and domains which are CoreCast capable advertise it through the mapping system. S transmits regular unicast packets towards non-Corecast capable domains. Furthermore, routers only keep the hash of one packet per stream at a time, so for each stream, a new chunk replaces the previous one. This ensures the scalability of the protocol. And finally, CoreCast can be extended by different means, for instance S could have read access to the LISP mapping system, and collaborate with xTRs, alleviating their state-load and improving the overall performance of our proposal.

4. REFERENCES

[1] M. Cha, et al. On next-generation telco managed P2P TV architectures. *IPTPS'08*, Feb. 2008.

[2] X. Hei, et al. A measurement study of a large-scale P2P IPTV system. *IEEE Transactions on Multimedia*, 9(8):1672–1687, Dec. 2007.

[3] D. Meyer. The locator identifier separation protocol (LISP). *The Internet Protocol Journal*, 11(1):23–36, 2008.