GoalBit: A Free and Open Source Peer-to-Peer Streaming Network

Andrés Barrios
InCo., Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

Matías Barrios
InCo., Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

Daniel De Vera
InCo., Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

Pablo Rodríguez-Bocca *
InCo., Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
prbocca@fing.edu.uy

Claudia Rostagnol
InCo., Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

SUBMITTED to ACM MULTIMEDIA 2011 OPEN SOURCE SOFTWARE COMPETITION.

ABSTRACT

This paper presents the GoalBit Starter platform. GoalBit is an open source peer-to-peer distribution system of real-time video streams over Internet. The main advantage of a P2P architecture is the possibility of using available upload bandwidth in the hosts connected. The main difficulty is that these hosts are typically highly dynamic, they continuously enter and leave the network. To deal with this problem a mesh connectivity approach is used (Bittorrent-like) where the stream is decomposed into several pieces and shared between different peers. Nowadays, the GoalBit platform is used by operators and by final users to broadcast their live contents. To illustrate its potential, we present some empirical results measured in an emulation of a GoalBit P2P streaming, with more than 300 peers concurrently connected.

Categories and Subject Descriptors

C.2.4 [**Computer-communication Networks**]: Distributed Systems

General Terms

Design, Performance, Measurement

Keywords

Video, Streaming, Peer-to-Peer

MM'11, November 28–December 1, 2011, Scotsdale, Arizona, USA.

1. INTRODUCTION

GoalBit is a free and open source peer-to-peer streaming network [4]. It provides the open source tools for broadcasting, distribute and watch live P2P video streamings. Live video P2P networks have to satisfy harder constraints than common P2P file–sharing system, mainly because video reproduction has real–time restrictions (file sharing do not) and because the nodes only remain connected to the network a few minutes [18]. Nowadays some commercial P2P networks for live video distribution are available. The most successful are PPlive [14] with more than $200,000$ concurrent users on a single channel [16] (reaching an aggregate bit-rate of 100 Gbps), SopCast [17] with more than $100,000$ concurrent users reported by its developers, CoolStreaming [6] with more than $25,000$ concurrent users on a single channel [24], PPstream [15], TVAnts [20], and TVUnetwork [21].

PPLive [14] is the most popular P2PTV software, especially for Asian users. Born as a student project in the Huazhong University of Science and Technology of China, PPLive uses a proprietary protocol, and its source-code is not available. With reverse engineering, [11] and [2] show that it uses a Bittorrent-like approach, with a channel selection bootstrap from a Web tracker, and peer download/upload video in chunks from/to multiple peers. It uses two kinds of video encoding formats, Window Media Video (VC-1) and Real Video (RMVB), with average bit rates from 250 Kbps to 400 Kbps, and it uses the user native media player to display the video. The distribution engine (Bittorrent-like protocol) does not use the *rarest-first* download policy, because the streaming must satisfy real–time restrictions. Also the *tit-for-tat* upload policy is not applied: when a client joins a channel, the other peers in the swarm send chunks forcefully to minimize the playback startup delay.

SopCast [17], created at the Fudan University of China, is very similar to the PPLive project. Following the same reverse engineering procedure, [11, 2] show that SopCast uses a Bittorrent-like approach, with a proprietary protocol very close to the PPLive protocol. Read [16] for a complete performance study of this network.

CoolStreaming [6] (also known as DONet) is the predecessor of PPLive and SopCast, now forced to close down due to copyright issues. Also using a proprietary protocol,

and closed source-code, as PPLive and SopCast, the CoolStreaming authors published information about its distribution engine [25, 23, 24]. The video stream is divided into chunks with equal size. A difference of CoolStreaming, with respect to PPLive and SopCast, is that the video stream is decomposed into six sub-streams by a rotating selection of video chunks. These commercial P2P networks implement a mesh–based distribution overlay, where there is not a defined topology and the list of peers connected is mantained through some method like the BitTorrent tracker [5] or Kademlia [12]. Its protocol applies more restrictions than a basic P2P sharing protocol for accomplishing natural requirements that live video reproduction has.

Another much more popular overlay in academic research is the tree–based one. In a P2P network with this overlay every peer is a node of a tree, receiving the stream from its parent and sending it to its child nodes. The possible childs every node can have influences the distance between the broadcaster (the first parent node) and the rest. The delay of the streaming is the most notorious undesired effect for the most far away nodes. This overlay also sensible to the disconnection of peers, un-providing its childs with the stream. Many systems are based in this overlay and have their own protocol for building and maintaining the multicast tree. For example, both NICE [3] and Zigzag [19] adopt hierarchical distribution trees and therefore scale to a large number of peers. SpreadIt [7] constructs a distribution tree rooted at the sender for a live media streaming session. A new receiver joins by traversing the tree starting at the root, until it reaches a node with sufficient remaining capacity. PeerCast [13] is a P2P network designed for live video and audio broadcast over Internet. Its protocol (based in the tree–based overlay) and source code is open–source and its usage is free for consumers and broadcasters.

In this paper we present the GoalBit platform. Its first difference with the previous mentioned commercial networks is that GoalBit is the first free and open source peer-to-peer platform with a mesh–based distribution overlay used by end-users on the Internet for real-time video streaming. Also, GoalBit is a complete suite for live media distribution, which implements the media generation, encapsulation, distribution and reproduction. For the generation, the live media can be captured using different kind of devices (such as a capture card, a webcam, another http/mms/rtp streaming, a file, etc.), it can be encoded to many different formats (MPEG2/4-AVC, VC-1, ACC, VORBIS, MPGA, WMA, etc.) and muxed into various containers (MPEG-TS, ASF, OGG, MP4, etc). The streaming encapsulation defined as GoalBit Packetized Stream (GBPS) takes the muxed stream and generates fixed size pieces (chunks), which are later distributed using the GoalBit Transport Protocol (GBTP). As we will explain later, the GBTP uses a Bittorrent-like approach. Finally (at the client side) the GoalBit player reproduces the video streaming consuming the pieces obtained through GBTP. For the reproduction we integrate the Videolan Media Player [22] into our source code, therefore the final users do not need to install an external media player software. Nowadays, GoalBit platform is used by operators (as ANTEL in their video portal AdinetTV [1] and by final users to broadcast their live contents.

The paper is organized as follows. Section 2 presents the global architecture of the GoalBit Starter platform, the software components needed and a description of their main functionalities. In Section 3 we introduce the GoalBit Transport Protocol. In Section 4 we show empirical results of an emulation of a GoalBit P2P streaming with more than 300 peers connected. Finally, the main contributions of this work are then summarized in Section 5.

2. SOFTWARE ARCHITECTURE

In a GoalBit P2P live video streaming three essential components must exist according to the GoalBit architecture, please read [10] for details. Firstly, the broadcaster, who creates the video streaming from some video source. Secondly, the tracker, who reports to the broadcaster and viewers the presence of the rest. Thirdly, the viewer, who receives the streaming and watches it. As of today, GoalBit provides the software for creating those three components.

Broadcasters and viewers must have the GoalBit Media Player installed in their hosts in order to send and receive video streams. GoalBit Media Player can be used embedded in a Web browser as a Web plugin (we also offer a small size GoalBit Media Player release with only the Web plugin interface). The Starter Suite is a Web server application that has a tracker incorporated and therefore enables viewers and broadcasters to make contact. Also, Starter Suite enables to configure a broadcaster via one of its web pages, and it easily shares the URL of the stream to viewers.

The main functionalities of the Starter Suite to the end-user are web broadcasting and channel management. When the user opens the Starter's Suite main page a list of the registered channels is shown, indicating the amount of viewers currently watching. A channel can be watched right there just by clicking its name. An URL for its sharing is given, along with HTML code for embedding it in another web page. If desired, a channel may be unregistered.

For web broadcasting and thus adding a new channel in the Starter Suite, the link named "Broadcast now!" at the top of the channel listing must be used. This opens a new web page where the channel and broadcaster are configured. The channel name must be specified, and then a playlist must be assembled for the broadcaster. It may contain video files, network streamings, CDs/DVDs, a webcam or a capture device as items of the playlist. Once the channel name and playlist are specified, the broadcast starts by clicking the button below the playlist. If the channel list is opened, the new channel is shown, along with the presence of the broadcaster. The creator sends then the channel's sharing URL to the viewers. They must open it using their web browsers with the Web Plugin previously installed. The channel name will be shown and also a play button to start displaying the video stream. The Web Plugin does not offer any other functionality than those offered indirectly through web pages, such as broadcasting and watching a GoalBit stream.

PHP, MySQL and JavaScript are the main technologies used for the Starter Suite. Its license is AGPL [8]. The Web Plugin is written in C and its source code is based on the VLC player's, plus the implementation of the GoalBit protocol and modifications for playing and broadcasting with it. Its license is GPLv2 [9].

3. GOALBIT PROTOCOL

As commented before the GoalBit Transport Protocol (GBTP) is an extended version of the BitTorrent Protocol, optimized for live video transmissions. In this section we introduce

some of its main characteristics and its specification, for a detailed information please read [4].

3.1 Basic Concepts

In GBTP (as in Bittorrent) each piece of the stream is identified by a unique number, called the piece ID. It is used to accomplish the exchange of pieces between peers. As GBTP is designed to distribute a continuous streaming and not a file, the piece numbering is cyclic in the $[0, MAX\_PIECE\_ID]$ interval[1].

Each peer has a sliding window (over previous interval of possible ID values) associated with it. The sliding window has a minimum value (called the *base*) and a length (which defines the maximum window value). Peers only can download or share pieces whose IDs are included in their respective windows

In order to reproduce the streaming, the peers have a player which consumes the pieces stored in their sliding windows in a sequential way. The *execution index* is defined as the next piece ID to be consumed by the player.

Other important concepts used in the protocol are the *active buffer* and the *active buffer index* (or ABI). The active buffer is defined as the consecutive sequence of pieces that a peer has, starting at the execution index. In other words, this corresponds to the downloaded video that the player will reproduce without interruption at a specific time. Finally, the active buffer index is the largest piece ID inside the active buffer.

3.2 Tracker-peer communication

There are two types of exchanges between tracker and peers: the initial one, and the exchanges that take place periodically during the sojourn of the peer in the network. These communications are carried out over the `HTTP/HTTPS` protocol and are based on a unique message called *announce*.

Initially, the peers contact the tracker in order to get a list of peers and an index from where to start asking for pieces. Normally, the peer sends an announce to the tracker with $numwant = 55$ and $ABI = MAX\_PIECE\_ID + 1$. In the answer, the peer gets a list of peers with at most *numwant* elements, the starting index and the peer type assigned.

Since the tracker needs to store status information for each peer, there must be a regular communication between them to allow this transfer of information. In the periodic communication (and leaving aside some particular cases), the peer sends an announce with $numwant = 0$, $ABI =$ the current ABI, and $QoE =$ the quality of experience measured by the peer in the last period. In this case, there is no relevant information in the tracker's answer. At those exchanges peers can also ask for more peers. These exchanges are regulated by the tracker, who defines the minimal time interval between two consecutive announces.

3.3 Peer-to-peer communication

Peers exchange messages using the TCP protocol. Two kinds of messages are exchanged. The first are used to inform about the contextual situation of each peer, i.e. which peer remains connected and which pieces already have been downloaded by every peer. To exchange contextual information five type of messages are used: HANDSHAKE, BITFIELD, HAVE, WINDOW UPDATE and KEEP-ALIVE. The other group of messages are used to exchange pieces. The process to exchange a piece is non trivial, and eight types of messages are used: INTERESTED, NOT INTERESTED, CHOKE, UNCHOKE, REQUEST, CANCEL, PIECE and DONT HAVE.

[1]Currently the MAX_PIECE_ID parameter is set to 2^{31}.

4. TESTS AND RESULTS

An emulation of a GoalBit P2P network sharing a single video streaming was run on November, 2010. All the servers used were run inside a cloud computing service. The emulation lasted for two hours, reaching a peak of 300 simultaneous viewers (executed as computer processes). During the emulation eight human viewers and a total of 340 emulated viewers participated of the streaming.

Different type of servers participated, none of them was overloaded in RAM and CPU resources. One server ran the Starter Suite and the GoalBit Media Player as broadcaster. Two servers ran up to twenty viewers with special upload capacity, forty viewers in total. Seventeen servers ran viewers with normal upload capacity with a peak of twenty each. Eight personal computers ran just one viewer. The source of the streaming was a file that had an average bitrate of 2000 kbps. Each normal viewer had an upload bandwidth limit equal to the half of the video bitrate, being 1000 kbps.

The quality of experience of every viewer was measured according to the initial buffering time, rebuffering time (time spent waiting for video to start again), chunk loss rate (chunks lost vs chunks received), and general failure (were not able to watch even a frame of the video).

The results for the emulation were as follow. The average initial buffering time was 5.55 seconds, the average rebuffering time was 11.83 seconds, the amount of peers that rebuffered: 15.88% (they only rebuffered once), the average chunk loss rate was 0.0006% and the average failed peers was 2.9%.

Figure 1 shows the percentage of peers in function of the seconds that buffering lasted at most. 50% of the peers lasted at most 5 seconds to start showing the video. 80% of the peers lasted at most 7 seconds.

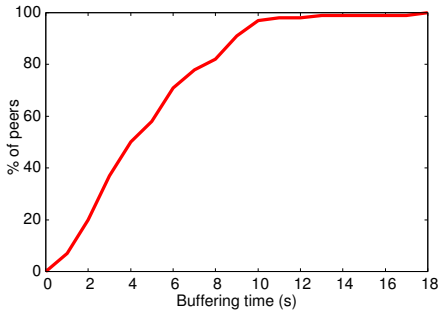

Figure 1: Cumulative distribution function of the buffering time.

Figure 2 shows the percentage of peers in function of the seconds that rebuffering lasted at most. 60% of the peers lasted at most 10 seconds to start watching the video again. 80% of the peers lasted at most 13 seconds.

None of the eight human viewers rebuffered and watched the video in its original conditions because no transcoding was applied to it.

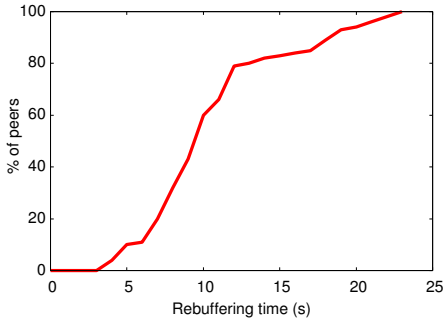

Figure 2: Cumulative distribution function of the rebuffering time.

The overhead is the extra amount of bit transference needed for the protocol. The global overhead was of 0.3293%. The average bandwidth saving was 59%. This means that viewers uploaded 59% of the total traffic.

5. CONCLUSIONS

This paper presents the GoalBit Starter platform, an open source peer-to-peer distribution system of real-time video streams over Internet. Nowadays, the GoalBit platform is used by operators and by final users to broadcast their live contents. Our empirical results, with more than 300 peers concurrently connected, showed a bandwidth saving of 59%, without lost of quality in the video playback and with an insignificant start-up delay.

People searching for a simple way of broadcasting using P2P technology should try this software. Organizations interested in sharing live video streamings in their LAN with a simple and easy to use starter platform will find this software useful. Organizations with live video streaming platforms over Internet trying to reduce bandwidth costs will find it useful too.

6. REFERENCES

[1] AdinetTV home page. http://www.adinettv.com.uy/, 2011.

[2] S. Ali, A. Mathur, and H. Zhang. Measurement of commercial peer-to-peer live video streaming. In *In Proc. of ICST Workshop on Recent Advances in Peer-to-Peer Streaming*, Weaterloo, Canadda, 2006.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 205–217, New York, NY, USA, 2002. ACM Press.

[4] M. E. Bertinat, D. D. Vera, D. Padula, F. Robledo, P. Rodríguez-Bocca, P. Romero, and G. Rubino. Goalbit: The first free and open source peer-to-peer streaming network. In *Proceedings of the 5th international IFIP/ACM Latin American conference on Networking (LANC'09)*, pages 83–93, New York, USA, September 2009. ACM.

[5] Bittorrent home page. http://www.bittorrent.org, 2007.

[6] CoolStreaming home page. http://www.coolstreaming.us, 2007.

[7] H. Deshpande, M. Bawa, and H. Garcia-Molina. Streaming live media over a peer-to-peer network. Technical Report 2001-30, Stanford InfoLab, April 2001.

[8] GNU Affero General Public License. Version 3. http://www.gnu.org/licenses/agpl-3.0.txt, 2007.

[9] GNU General Public License. Version 2. http://www.gnu.org/licenses/gpl-2.0.txt, 1991.

[10] GoalBit Architecture. http://goalbit.sourceforge.net/architecture.html, 2009.

[11] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. Insights into pplive: A measurement study of a large-scale p2p iptv system. In *In Proc. of IPTV Workshop, International World Wide Web Conference*, 2006.

[12] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 53–65, London, UK, 2002. Springer-Verlag.

[13] Peercast home page. http://www.peercast.org/, 2007.

[14] PPLive Home page. http://www.pplive.com, 2007.

[15] PPStream home page. http://www.ppstream.com/, 2007.

[16] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari. Will iptv ride the peer-to-peer stream? *Communications Magazine, IEEE*, 45:86–92, 2007.

[17] SopCast - Free P2P internet TV. http://www.sopcast.org, 2007.

[18] K. Sripanidkulchai, B. Maggs, and H. Zhang. An analysis of live streaming workloads on the internet. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 41–54, New York, NY, USA, 2004. ACM Press.

[19] D. A. Tran, K. A. Hua, and T. T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. In *INFOCOM*, 2003.

[20] TVAnts home page. http://cache.tvants.com/, 2007.

[21] TVUnetworks home page. http://tvunetworks.com/, 2007.

[22] VideoLan home page. http://www.videolan.org, 2007.

[23] S. Xie, G. Y. Keung, and B. Li. A measurement of a large-scale peer-to-peer live video streaming system. In *ICPPW '07: Proceedings of the 2007 International Conference on Parallel Processing Workshops (ICPPW 2007)*, page 57, Washington, DC, USA, 2007. IEEE Computer Society.

[24] X. Zhang, J. Liu, and B. Li. On large scale peer-to-peer live video distribution: Coolstreaming and its preliminary experimental results. In *IEEE International Workshop on Multimedia Signal Processing (MMSP'05)*, Washington, DC, USA, 2005. IEEE Computer Society.

[25] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming. In *IEEE Conference on Computer Communications (INFOCOM'05)*, Washington, DC, USA, 2005. IEEE Computer Society.