G.S.PAVANENDRA, V. Sneha latha / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 3, May-Jun 2012, pp. 390-394

A NOVEL APPROACH LIVE STREAMING USING COOL STREAMING

G.S.PAVANENDRA*1, V. Sneha latha*2
*1 Student, M.Tech (CSE) ,Vaddeswaram,Green Fields, KL University, Vijayawada , Andhra Pradesh, India
*2 Assistant Professor, DEPT of CSE, KL University, Vaddeswaram , Vijayawada , Andhra Pradesh, India

Abstract –
A number of commercial peer-to-peer systems for live streaming have been introduced in recent years. The behavior of these popular systems has been extensively studied in several measurement papers. Due to the proprietary nature of these commercial systems, however, these studies have to rely on a "black-box" approach, where packet traces are collected from a single or a limited number of measurement points, to infer various properties of traffic on the control and data planes. Although such studies are useful to compare different systems from end-user's perspective, it is difficult to intuitively understand the observed properties without fully reverse-engineering the underlying systems. In this paper we describe how cool streaming are used and We emphasize three salient features of this data-driven design: 1) easy to implement, as it does not have to construct and maintain a complex global structure; 2) efficient, as data forwarding is dynamically determined according to data availability while not restricted by specific directions; and 3) robust and resilient, as the partnerships enable adaptive and quick switching among multi-suppliers.

Index Terms— IPTV, measurement, peer-to-peer technology, video streaming, live streaming, network architecture.

I. INTRODUCTION

There is an emerging market for IPTV. Numerous commercial systems now offer services over the Internet that
are similar to traditional over-the-air, cable, or satellite TV. Live television, time-shifted programming, and content-on demand are all presently available over the Internet. Increased broadband speed, growth of broadband subscription base, and improved video compression technologies have contributed to the emergence of these IPTV services.

We draw a distinction between three uses of peer-to-peer (P2P) networks: delay tolerant file download of archival material, delay sensitive progressive download (or streaming) of archival material, and real-time live streaming. In the first case, the completion of download is elastic, depending on available bandwidth in the P2P network. The application buffer receives data as it trickles in and informs the user upon the completion of download. The user can then start playing back the file for viewing in the case of a video file. Bit torrent and variants are example of delay-tolerant file download systems. In the second case, video playback starts as soon as the application assesses it has sufficient data buffered that, given the estimated download rate and the playback rate, it will not deplete the buffer before the end of file. If this assessment is wrong, the application would have to either pause playback or rebuffed, or slow down playback. While users would like playback to start as soon as possible, the application has some degree of freedom in trading off playback start time against estimated network capacity. Most video-on demand systems are examples of delay-sensitive progressive download application. The third case, real-time live streaming, has the most stringent delay requirement. While progressive download may tolerate initial buffering of tens of seconds or even minutes, live streaming generally cannot tolerate more than a few seconds of buffering. Taking into account the delay introduced by signal ingest and encoding, and network transmission and propagation, the live streaming system can introduce only a few seconds of buffering time end-to-end and still be considered "live" [1].

The Zattoo peer-to-peer live streaming system was a free to- use network serving over 3 million registered users in eight European countries at the time of study, with a maximum of over 60,000 concurrent users on a single channel.
Cool streaming, a *data-driven* scheme, is a completely different approach [2]. The key novelties are 1) peers gossip with one another for content availability information and 2) peers use a swarm-like technique, somewhat similar to the technique used in Bit Torrent, for content delivery. We make a distinction in this paper by referring this as a peer-to-peer (P2P) live video streaming system, in which there is no explicit overlay topology construction. This was referred to as a treeless approach or swarming in [3]. We refer to the other approaches as overlay multicast, given the explicit construction of multicast tree(s).

Cool streaming represented one of the earliest large-scale P2P video streaming experiments [4], which has been widely referenced in the community as the benchmark (Google entries tops 300 000) that a P2P-based live streaming system has the potential to scale. Since its first release, while keeping the random partner selection, we have enhanced the system in nearly all aspects, specifically 1) the initial system adopted a simple pull based scheme for content delivery based on content availability information exchange using buffer-map. This incurs per block overhead, and more importantly, it results in a longer delay in retrieving the video content. We have unimplemented a hybrid pull and push mechanism, in which the video content is pushed by a parent node to a child node except for the first block. This lowers the overhead associated with each video block transmission, reduces the initial delay and increases the video playback quality; 2) a novel multiple sub stream scheme is implemented, which essentially enables multisource and multipath delivery for video streams. Observed from the results, this not only enhances
the video playback quality and but also improves the effectiveness against system dynamics; 3) the gossip protocol was enhanced to handle the push function; 4) the buffer management and scheduling schemes are redesigned to deal with the dissemination of multiple sub streams.

There are many issues relevant to the design of live video streaming systems such as system dynamics, heterogeneity, churn, network congestion, stream synchronization, multipath routing, topology stability and convergence [5]–[6]. There have been investigations on optimal resource allocation scaling factors [7], incentive-based mechanism fine-granularity control [8], priority based multicast [9], integration with Network encoding technique [10]. However, there seems to be several misconceptions in the most basic understanding of a live video streaming system. This paper takes a different approach by contributing to this basic understanding, we will: 1) describe the key issues in a real working system; 2) closely examine a set of existing practical problems and how they could be handled in a real system; 3) focus the discussions on system dynamics and how it affects the overall performance.

Recall one of the fundamental principles in the Internet is the simplicity in its core functionalities. Keeping this simplicity in mind, our discussions in this paper focus on a reasonably scale no optimal working system. The purpose of this paper is to demonstrate concretely how a working system resolves some of these issues and a set of realistic engineering problems that need to be addressed.

II. RELATED WORK

In this section we present work that used in live streaming it is mainly used in IPTV [1]. Hyunseok chang, paper elaborates live streaming uses zattoo network rebroadcast live TV in internet .[2] Xinyan Zhang, paper elaborates cool streaming in overly network for efficient live media streaming .[3] Susu Xie, paper elaborates cool streaming Design, Theory, and Practice.

Based above study we proposed new system for "live streaming using cool streaming" mainly used live streaming by using zattoo network, Zattoo network receive signal from satellite and it will broadcast to multiple peers. Cool streaming focused on the efficiency associated with multitree construction and also explored the simplicity and scalability adopted from unstructured networks.

Live streaming is main used watching TV or video without buffing in internet, live streaming [4] uses zattoo system rebroadcast live TV, it captured from satellites on the internet, and this system carries each TV channel from separate peer to peer network. Every user first register and download application from internet, user get ticket limited amount of time.

III SYSTEM ARCHITECTURE

Live streaming uses zattoo because of rebroadcasting (if any link fails then it will rebroadcast) live TV, captured from satellites, on the internet. The zattoo carries TV channel on separate peer-to-peer delivery network. It can carries limited number of TV channel. Fig 1 show a typical setup of a single TV channel carried on the zattoo network. The TV signal captured from satellite is encode or encrypted, and send onto zattoo network. Users are required to register themselves at the zattoo website to download the zattoo player application. To receive the signal of a channel, the users first authenticate itself to the zattoo authentication server. Authentication means user is granted a ticket with limited amount of time. The ticket specifies that the user is authorized to receive TV signal. Zattoo [11] uses the reed-Solomon error correcting code for forward error correction. The RS code is a systematic code: of the n packets sent per segment, $k < n$ packets carry the live stream data while the remainder carries the redundant data. Due to the variable-bit rate nature of the data stream, the time period covered by a segment is variable, and a packet may be of size less than the maximum packet size. A packet smaller than the maximum packet size is zero-padded to the maximum packet size for the purposes of computing the (shortened) RS code, but is transmitted in its original size. Once a peer has received k packets

per segment, it can reconstruct the remaining n – k packets.

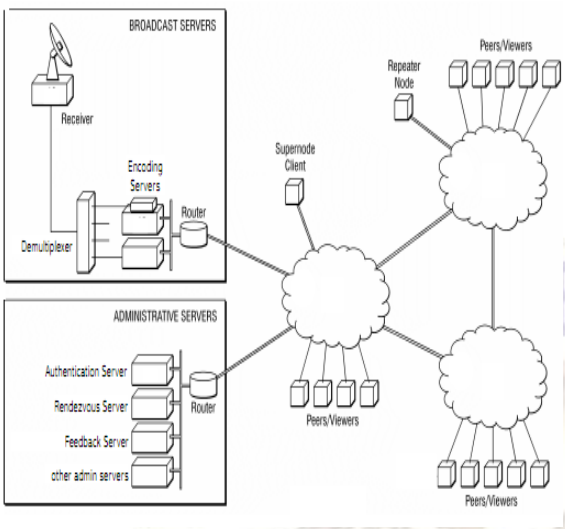

Fig 1 Zattoo delivery network architecture.

A. *Why we use cool streaming*

Cool streaming is mainly used for:

- Easy to implement, as it does not have to construct and maintain a complex global structure.
- Efficient, as data forwarding is dynamically determined according to data availability while not restricted by specific directions.
- Robust and resilient, as the partnerships enable adaptive and quick switching among multi-suppliers.

B. *Basic Components*

There are three basic modules in the system: 1) Membership manager, which maintains partial view of the overlay. 2) *Partnership manager*, which establishes and maintains TCP connections, or partnership, with other peer nodes. It also exchanges the availability of stream data in the *buffer map* (BM) with the peer nodes, which we will explain later. 3) *Stream manager*, which is the key component for data delivery. Besides providing stream data to the media player, it also makes decisions on where and how to retrieve stream data. The central design in this system is based on the *data-driven* notion, in which every peer node periodically exchanges its data availability information with a set of partners to retrieve unavailable data, while also supplying available data to others. Fig 2 shows how cool streaming is working.

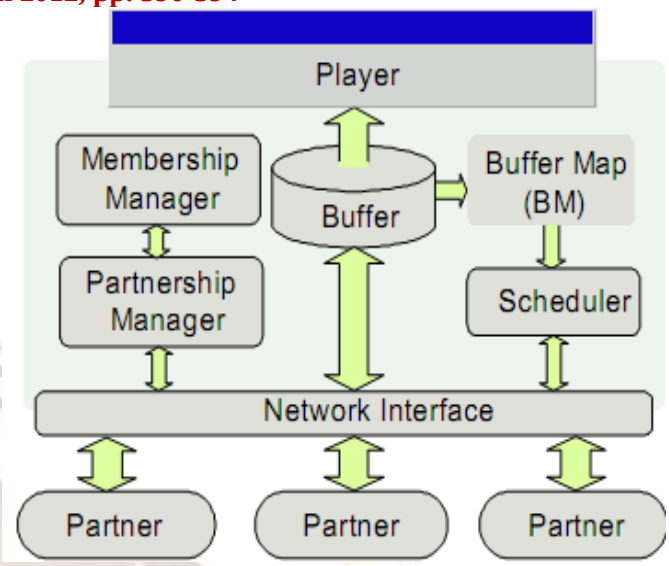

Fig 2 Cool streaming system diagram.

C. *Multiple Sub streams*

The video stream is divided into *blocks* with equal sizes, and each block is assigned a sequence number to represent its playback order in the stream. Since it is a live video streaming and we use TCP for transmissions, the sequence number also serves as a timestamp, which can be used to combine and reorder the blocks after reception. One of the key factors contributing to the success in P2P file sharing applications is the adoption of the gossip concept, in which a node can request different small chunks of file content from different nodes. This achieves significantly higher efficiency compared to other traditional systems. This is also adopted in cool streaming. Specifically, a video stream is divided into multiple *sub streams*; nodes could subscribe to different sub streams from different partners.

1) **End-to-end bandwidth is a key problem in large scale streaming:** In a client/server system, we can support a large number of users in a local area by adding more servers. For a global scaled service, simply adding servers is not enough however. The end-to-end bandwidth may limit the geographical distribution of the users. Hence, CDN is a possible solution, but it remains very expensive and is not readily deployable. On the other hand, P2P enables intelligent path selections that may avoid the above problem. Nonetheless, we also find that, comparing with client-server solution, the overlay solution may introduce additional delay for a user to smoothly playback the video.
2) **Upload bandwidth is a physical limitation:** While P2P streaming is more flexible than client/server, its does have certain limitations. The

most significant is the demands on the upload bandwidth. For a successful P2P media streaming system, at least we should have average upload bandwidth larger than the streaming rate. Hence, ADSL and other asymmetrical Internet accesses become obstacles.

3) **ISP issues and traffic engineering effects:** Currently, a large portion of the available bandwidth at network edges and backbone links are occupied by such P2P applications as BitTorrent. Many ISPs thus limit this kind of traffic. For file sharing the limitation may only be annoying (just slow down the download),

but for media streaming it can be fatal. Other filtering mechanisms may also create problem to p2p streaming systems. We have already observed such problems in CoolStreaming.

D. Tree-based Protocols and Extensions

As mentioned previously, many overlay streaming systems employ a tree structure, stemmed from IP multicast. Constructing and maintaining an efficient distribution tree among the overlay nodes is a key issue to these systems. In Coop Net [3], the video source, as the root of the tree, collects the information of all the nodes for tree construction and maintenance. Such a centralized algorithm can be very efficient, but relies on a powerful and dedicated root node. To the contrary, distributed algorithms, such as Spread It, NICE [12], and ZIGZAG [11], perform the constructing and routing functions across a series of nodes. For a large-scale network, these algorithms adopt hierarchical clustering to achieve minimized transmission delay (in terms of tree height) as well as bounded node workload. Still, an internal node in a tree has a higher load and its leave or crash often causes buffer underflow in a large population of descendants. Several tree repairing algorithms have been devised to accommodate node dynamics yet the tree structure may still experience frequent breaks in the highly dynamic Internet environment.

E. *Buffering*

A *buffer map* or BM is introduced to represent the availability of the latest blocks of different sub streams in the buffer. This information also has to be exchanged periodically among partners in order to determine which sub stream to subscribe to. The detailed structure of the buffer map is as follows: BM is represented by a series 2K of -tuples, where K is the number of sub streams.

IV. CONCLUSIONS

We proposed new system for live streaming using cool streaming. Cool streaming focused on the efficiency associated with multitree construction and also explored the simplicity and scalability adopted from unstructured networks. There are two points that we like to emphasize from this study: 1) a working system is essential in providing basic understanding; 2) there is a large number of practical problems that have to be dealt with in real engineering.

V. ACKNOWLEDGMENTS

We are greatly delighted to place my most profound appreciation to Er.K.Satyanaraysana Chancellor of K.L.University, Assistant Prof V. Sneha latha Guide, Dr.K.Raja Sekhara Rao Principal, Prof S.Venkateswarlu Head of the department, and Dr.Subramanyam in charge for M.Tech under their guidance and encouragement and kindness in giving us the opportunity to carry out the paper. Their pleasure nature, directions, concerns towards us and their readiness to share ideas enthused us and rejuvenated our efforts towards our goal. We also thank the anonymous references of this paper for their valuable comments.

REFERENCES

1. S. Xie, B. Li, G. Y. Keung, and X. Zhang, "CoolStreaming: Design, Theory, and Practice," *IEEE Trans. on Multimedia*, vol. 9, no. 8, December 2007.
2. Hyunseok chang, "live Streaming with receiver base peer division multiplexing", in *proc. IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 19, NO. 1, FEB 2011*
3. S. Q. Zhuang, B. Y. Zhao, and A. D. Joseph, "Bayeux: An architecturefor scalable and fault-tolerant wide-area data dissemination," in *Proc. NOSSDAV'01*, New York, Jun. 2001.
4. D. Tran, K. Hua, and T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming," in *Proc. IEEE INFOCOM*, 2003.
5. L. Guo, S. Chen, S. Ren, X. Chen, and S. Jiang, "PROP: a scalable and reliable P2P assisted proxy streaming system," in *Proc. ICDCS'04*, Tokyo, Japan, Mar. 2004.
6. V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proc. NOSSDAV'02*, USA, May 2002.
7. S. Jin and A. Bestavros, "Cache-and-relay streaming media delivery for asynchronous clients," in *Proc. International Workshop on Networked Group Communication (NGC'02)*, Boston, MA, USA, Oct. 2002.
8. W. Wang and B. Li, "Market-based self-optimization for autonomic service overlay

networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 12, Dec. 2005.

9. M. Wang and B. Li, "Lava: A reality check of network coding in peer-to-peer live streaming," in *Proc. IEEE Infocom*, May 2007.
10. Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: peer-to-peer patching scheme for VoD service," in *Proc. WWW'03*, Budapest, Hungary, May 2003.
11. D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming," in *IEEE J. Select. Areas in Comm.*, vol. 22, Jan. 2004.
12. N. Magharei and R. Rejaie, "PRIME: Peer-to-Peer Receiver-drIven MEsh-based Streaming," in *Proc. IEEE INFOCOM*, May 2007.
13. M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," in *Proc. ACM SOSP'03*, New York, USA, Oct. 2003.
14. N. Magharei, R. Rejaie, and Y. Guo, "Mesh or Multiple-Tree: A Comparative Study of Live P2P Streaming Approaches," in *Proc. IEEE INFOCOM*, May 2007.
15. A. J. Ganesh, A.-M. Kermarrec, and L. Massoulie, "Peer-to-peer membership management for gossip-based protocols," *IEEE Trans. Comput.*, vol. 52, no. 2, Feb. 2003.
16. M. Hefeeda *et al.*, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast," in *Proc. ACM Multimedia*, November 2003.
17. Y. Chawathe, S. McCanne, and E. A. Brewer, "An architecture for Internet content distribution as an infiastmcture service," http://yatin.chawathe.com/-yafin/papers/scattercast.ps.
18. Y.-W. Sung, M. Bishop, and S. Rao, "Enabling Contribution Awareness in an Overlay Broadcasting System," in *Proc. ACM SIGCOMM*, 2006.