LiveSky: Enhancing CDN with P2P

HAO YIN, XUENING LIU, TONGYU ZHAN, Tsinghua University
VYAS SEKAR, Carnegie Mellon University
FENG QIU, Beijing Blue I.T. Technologies Co., Ltd
CHUANG LIN, Tsinghua University
HUI ZHANG, Carnegie Mellon University
BO LI, Hong Kong University of Science and Technology

We present the design and deployment experiences with *LiveSky*, a commercial hybrid CDN-P2P live streaming system, which inherits the best of both CDN and P2P. We address several key challenges, including: 1) ease of integration with existing CDN infrastructure, 2) dynamic resource scaling while guaranteeing quality-of-service, 3) providing good user experience, ensuring network friendliness and upload fairness. *LiveSky* has been used for several large-scale live streaming events in China. Our evaluation results from real-world indicate that such a hybrid CDN-P2P system provides quality and performance comparable to a CDN and effectively scales the system capacity.

Categories and Subject Descriptors: C.2.4 [**Computer Communication Networks**]: Distributed Systems—*Distributed applications*

General Terms: Design, Measurement, Performance

Additional Key Words and Phrases: Content delivery networks, peer-to-peer, live streaming

ACM Reference Format:

Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., and Li, B. 2010. LiveSky: Enhancing CDN with P2P. ACM Trans. Multimedia Comput. Commun. Appl. 6, 3, Article 16 (August 2010), 19 pages.
DOI = 10.1145/1823746.1823750 http://doi.acm.org/10.1145/1823746.1823750

1. INTRODUCTION

Live video streaming has long been projected as the killer application for the Internet. While this expectation has been in effect for several years now, only in recent years with the deployment of

This work was funded by Project 60873254 supported by NSFC, Project 20090460317 supported by China Postdoctoral Science Foundation, and the Tsinghua-ChinaCache CDN Research Institute Project. V. Sekar and H. Zhang were supported in part by NSF award ANI-0331653. B. Li's research was supported in part by grants from RGC under the contracts 615608 and 616207, and by a grant from NSFC/RGC under the contract N_HKUST603/07.
The preliminary result of this work was presented at ACM Multimedia 2009.
Authors' addresses: H. Yin (corresponding author, email: h-yin@tsinghua.edu.cn). X. Liu, T. Zhan, and C. Lin, Department of Computer Science and Technology, Tsinghua University, Beijing, China; V. Sekar and H. Zhang, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA; F. Qiu, Research and Development Department, Beijing Blue I.T. Technologies Co., Ltd, Beijing, China; B. Li, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China.

DOI 10.1145/1823746.1823750 http://doi.acm.org/10.1145/1823746.1823750

increased bandwidth in the last-mile has this promise finally turned into reality [Kirkpatrick 2008; Gannes 2009].

Content Delivery Networks (CDNs) and Peer-to-Peer (P2P) systems are two alternative and competing technologies for delivering Internet live streaming. CDNs, such as Akamai and Limelight, deploy servers in multiple geographically diverse locations, distributed over multiple Internet Service Providers (ISPs). User requests are redirected to one of the best available servers based on proximity, server load, and so on. CDNs serve end users with the appearance of traditional client-server approaches, but enable content providers to handle much larger request volumes. Thus, end users observe higher quality experience and content providers can offer more reliable service. At the same time, ISPs can also benefit from deploying CDN servers in their networks as it reduces the total amount of upstream and transit traffic. On the other hand, P2P technologies solve the scalability issue by leveraging the resources of the participating peers. Several P2P video streaming systems have attracted a large number of Internet viewers, as demonstrated by the huge popularity of applications such as PPLive, Joost, UUSee, CoolStreaming, etc. For example, PPLive used less than 10 Mbps of server bandwidth to simultaneously serve a 400 kbps video stream to roughly 1.5 million end users [Huang 2007].

Both approaches have their advantages and disadvantages. CDNs provide excellent quality to end users when the workload is within the provisioning limits. CDNs typically have to provision servers and bandwidth in advance using estimates of the expected workload and are thus inherently constrained by the specifics of their operating regime. Anecdotal evidence, for example, during the recent U.S. presidential inauguration [Vance 2009], suggests that even popular sites can be overwhelmed by unexpected surges in demand and thus have to deny service to end users. This scaling constraint becomes especially relevant as end users and content providers demand higher quality video, that is, implying higher operating costs for the CDNs.

Though P2P systems achieve high scalability while keeping the server requirements low, the decentralized and uncoordinated operation implies that this scaling comes with undesirable side effects. The data in live streaming is very time-sensitive. Delayed arrivals of data packets, probably caused by network congestions, retransmissions for packet losses, or peer dynamics in P2P overlay, will lead to rebuffering and playback discontinuities on the client side. P2P client usually uses a larger-size buffer to smooth the instability of data transmissions to provide a fluent viewing experience. However this will lead to higher delays, including the startup delay and the source-to-end delay. Moreover, previous measurement and analysis studies have denoted that there are several significant problems in existing P2P systems, including: 1) low stream quality with undesirable disruptions [Wu et al. 2009], 2) unfairness in the face of heterogeneous peer resources, for example, high-bandwidth peers become heavily loaded, but peers behind Network Address Translations (called NAT-ed hosts) do not contribute [Li et al. 2007]; and 3) network unfriendliness [Ali et al. 2006], that is, large numbers of the P2P traffic are across multiple ISPs or ASes (Autonomous Systems), even across multiple continents; however, the ISPs cannot profit from these P2P traffic. At the same time, it has been reported that certain access ISPs have surreptitiously blocked their customers from uploading P2P data using the popular P2P protocols [Dischinger et al. 2008].

A natural question is if we can build a hybrid CDN-P2P architecture that incorporates the best of both technologies and mutually offsets each others' deficiencies. Several researchers have hypothesized and analyzed the potential benefits of such an approach [Xu et al. 2006; Huang et al. 2008]. Other researchers have also recommended the use of hybrid approaches [Karagiannis et al. 2005; Pakkala and Latvakoski 2005; Rodriguez et al. 2006; Darlagiannis et al. 2007]. These works are basically via simulations and trace-driven analysis. However, we are not aware of any other reported works of real

implementation and deployment at a large scale that actually demonstrate the benefits of such a hybrid approach.

In this work, we present the design, implementation, real-world deployment and evaluation of *LiveSky*, a hybrid CDN-P2P live streaming system, developed and deployed by ChinaCache, one of the biggest CDN providers in China. We have addressed several key challenges in designing and deploying LiveSky, including: 1) ease of integration of the hybrid CDN-P2P system with existing CDN infrastructure, 2) a mechanism for dynamic resource scaling and allocation that guarantees adequate quality of service (QoS) to end users, 3) providing good user experience, including low startup delay, source-to-end delay, playback disruption rate, and so on, and 4) addressing the well-known shortcomings of P2P streaming system including unfairness in upload contributions and network unfriendliness.

An independent and equally valuable contribution of this paper is our analysis of the performance of LiveSky "in the wild." Several commercial providers have been recently adopting hybrid solutions to scale content distribution (e.g., RedSwoosh,[1] Octoshape[2]). However, real-world data on such deployments are hard to obtain. In this light, our evaluation becomes especially valuable for the future development of live streaming technologies and applications. Our measurement is based on an important live streaming event that LiveSky has been used to broadcast the stream in China. During this event, LiveSky served 400 kbps MPEG4-encoded video to more than 145,000 concurrent users. Our measurement results indicate that: 1) LiveSky works well even when the client upload bandwidth is restricted and effectively leverages the resources of both CDN and P2P nodes; 2) most clients obtain good quality service and suffer few viewing disruptions, even with a small 15 second client buffer; 3) users see more than two times shorter startup latency compared to pure P2P approaches; 4) LiveSky provides good performance even when the churn—the fraction of peers that join or leave in a specific interval (e.g., 1 minute)—is as high as 10%, effectively insulating most clients from churn-induced disruptions; and 5) LiveSky effectively offsets the deficiencies of P2P systems. Specifically, LiveSky leverages CDN redirections to make P2P transfers "network-friendly." Also, the presence of a publicly addressable and available CDN node enables NAT traversal, thereby better utilizing the available upload capacity of NAT-ed hosts.

To summarize, the key contributions of this article are, 1) design and implementation of LiveSky—a hybrid CDN-P2P system for effectively scaling the capacity of a CDN without compromising the reliability and user experience, 2) addressing key challenges in integrating the P2P component into the CDN, designing an adaptive scaling mechanism to guide the hybrid CDN-P2P operation, and overcoming the shortcomings of traditional P2P systems for live streaming, and 3) measurements and analysis of real-world deployments that validate the benefits of a hybrid CDN-P2P architecture.

The rest of the article is organized as follows. We present an overview of design and implementation of LiveSky system in Section 2 and give a more detailed discussion of the specific adaptations and optimizations of integrating CDN and P2P technologies in Section 3. Section 4 shows our real-world deployment and measurement results. We discuss the recent system enhancement and highlighted the future directions in Section 5 before we conclude this work in Section 6.

2. SYSTEM OVERVIEW

In this section, we describe the general architecture of *LiveSky*. As shown in Figure 1, it has three major components: (1) *Management center*, comprising the DNS-based Global Server Load Balance (GSLB) system that redirects user requests to the nearest, lightly loaded server, content management

[1]RedSwoosh, http://www.akamai.com/client/
[2]Octoshape, http://www.octoshape.com/

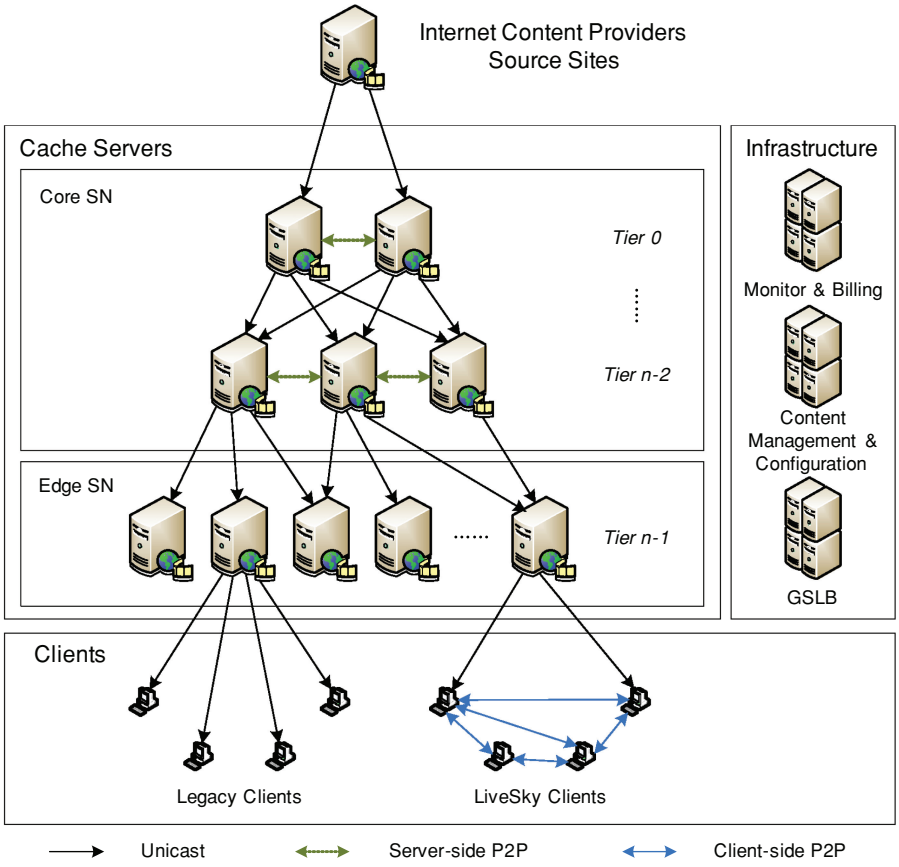


Fig. 1. System architecture.

and configuration systems, and monitoring and billing systems, which are responsible for efficient controlling and monitoring of the system. (2) *Cache servers*, referred to as Service Nodes (SNs) that deliver video contents from content providers to end users. The SNs are organized into several tiers (from Tier_0 to Tier_{n-1}) according to the scale demand, meanwhile satisfying the QoS and reliability requirement. For some practical cases, we may adopt n = 3 as Akamai do. We refer to the SNs in Tier_{n-1} as *edge* SNs since they are directly responsible for serving end users, and the remaining tiers are *core* SNs. The goal of the server side overlay is efficient data distribution with some measures to guard against some node failures and network delays. Since the SNs are more stable, the server-side distribution overlay is largely tree-based. However, in order to provide greater reliability in the presence of node or network failures, we allow each SN to retrieve the content either from SNs higher up in the hierarchy (i.e., a lower-numbered tier) or from peer SNs in the same tier. Since the edge SNs are responsible for serving end users, they are typically heavily loaded and we disable peering between SNs in the edge tier. Moreover, in order to ensure the stability of stream source, multiple source sites simultaneously offer the video stream to the Tier_0 SNs during the large-scale live event. (3) *End hosts*, which may either be legacy clients that directly obtain the stream from the edge SNs, or LiveSky-enabled clients that can additionally engage in P2P transfers.

2.1 System Operation

Before accessing the stream, a client first obtains the URL for the live stream from the content source (e.g., `livesky://domainname/live`). The GSLB component of the CDN takes into account the client location, the edge SN location, and the edge SN loads to find a suitable edge SN for this client. The client is then redirected to this edge SN using traditional DNS-based redirection techniques.

Each edge SN serves multiple roles acting as 1) a *regular server* for legacy clients, 2) a *tracker* for the P2P operation to bootstrap new clients with candidate peers, and 3) a *seed* for the P2P operation for the LiveSky-enabled clients assigned to it. The edge SNs are preconfigured with some decision logic that decides if a new LiveSky-enabled client should be served in CDN-mode or if they should be redirected to the P2P overlay (We will discuss this decision in detail in Section 3.1). Finally, the edge SN is used for some optimizations in the P2P operation. Note that the P2P overlays are localized on a per-edge SN basis; that is, the peers with which a LiveSky-enabled node communicates in the P2P mechanism are also assigned to the same edge SN as this node. The monitoring and GSLB systems periodically collect the operation status of each edge SN. The GSLB component redirects the clients requests to one of other nearly available servers when detecting the server node or network failures, thus avoiding the shutdown or network failures of one edge SN degrading the service stability in a per-edge communication group.

2.2 Client Side Content Distribution

Legacy Clients. There are two types of clients: legacy clients that receive contents directly from the edge SNs and LiveSky-enabled clients that can either receive contents from the edge SNs or additionally use P2P mechanisms. In practice, some users dislike installing any P2P client and plug-in software. Thus, our live streaming platform also provides service to these users (i.e., legacy clients) via traditional client-server approach. In order to encourage users to install the LiveSky client software and widespread adoption, the system may offer a higher quality video stream to LiveSky-enabled clients but only offer a lower quality stream to legacy clients during the same events. However, to demonstrate the effectivity of adopting hybrid CDN-P2P approach, we only focus on the LiveSky-enabled clients in our remaining design and evaluation discussions.

LiveSky's P2P Mechanism. There are two candidate P2P streaming technologies: tree-based [Chu et al. 2000] and mesh-based [Zhang et al. 2005]. Tree-based schemes provide low source-to-end delay, however, single-tree based overlays have poor performance under churn and fail to leverage the available upload capacity of peers effectively since majority of the peers are leaf nodes and contribute no capacity [Liu et al. 2008]. Multitree constructions alleviate some of these concerns [Castro et al. 2003; Venkataraman et al. 2006]. In contrast to the tree-based approach, mesh-based overlays are inherently robust to churn and also result in a more equitable use of uploading capacities of peers. However, mesh-based overlays may not deliver the relevant video segments in time and/or have to typically use large buffers to offset the effects of having to wait for available video segments to render. None of these approaches completely solve the problems arising from the dynamic operating environment, motivating recent proposals for hybrid P2P approaches combining the multi-tree and mesh schemes [Xie et al. 2007; Wang et al. 2007]. We adopt a similar scheme in LiveSky. The video stream is a single bit-rate encoding (i.e., we do not use any special or layered coding). In P2P transmission overlay, the stream is separated into several substreams according to the stream frame id. For example, if the video is divided into six substreams, substream_0 consists of frames $0, 6, 12, 18, \ldots$, substream_1 consists of frames $1, 7, 13, 19, \ldots$, and so on. Peers are organized in a tree-based overlay on a per-substream basis. Each peer has a parent node for each substream and obtain the stream frames belonging to this substream from the parent node. Additionally, in order to be robust to network

Table I. Notation Used in Our Analytical Model

Notation	Definition
ρ	The average fraction of full streaming rate that each supplying peer contributes during a session
k	Level in the P2P overlay; peers in lower numbered levels receive the content earlier and serve it to peers in the succeeding level. Especially, peers in level 1 directly receive the content from the CDN server
K_0	Maximum number of levels
N_c	CDN server capacity
P_0	Total number of clients for this media
S	Scaling factor, the fraction of the clients directly served by the CDN, S=N_c/P_0
N(k)	Upload capacity of clients in level k , $N(0) = N_c$
P(k)	Number of clients in levels $(k+1)\ldots K_0$, P(0)=P_0
λ	Join rate expressed as a fraction of peers that join in a specific interval
σ	Leave rate expressed as a fraction of peers that leave in a specific interval

or node failures, peers also construct a mesh-style overlay to retrieve missing frames for continuous playback.

3. ADAPTATION AND OPTIMIZATIONS

In this section, we present a more detailed discussion of two key aspects of the system: 1) system adaptation in the presence of unexpected load to achieve an efficient tradeoff between user quality, system cost and scalability, and 2) optimizations to address some of the shortcomings of P2P for effective live video delivery.

3.1 Adaptive Scaling

There are natural tradeoffs in a hybrid CDN-P2P environment between the operating cost of the CDN, the perceived user quality (e.g., delay between the video source and the end user), and the overall number of end users that can be supported by the system at a given time. The key challenge is to adapt the hybrid CDN-P2P operation depending on the working environment so that the users observe good performance and at the same time the operating costs of the CDN are not too high. To this end, we present an analytical model to understand the tradeoffs involved. The objective of the analysis is to guide the operation of the LiveSky system, especially to control the number of end users served directly by the CDN.[3]

The working environment for a hybrid CDN-P2P system is defined by several important parameters such as: 1) the media playback bitrate, 2) the total number of end users, 3) the bandwidth capacity of the edge SNs, 4) bounds on user quality defined as the delay between the video source and the end user, and 5) characteristics of end users such as their upload bandwidth capacity and churn rates. Given the working environment, the control parameter is the number of end users that are served directly by the edge SNs. Our analysis models the relationships between these parameters in order to derive guidelines for system operation.

Assumptions. In order to make the analysis tractable, we make four assumptions.

—First, we model the P2P overlay as a single tree, to avoid the complexity of modeling the hybrid multitree and mesh mechanism. Most of the time, this is a *conservative* assumption that underestimates

[3]Outreach [Small et al. 2007] has presented a similar work. However, the goals of two models are different. Outreach seeks the optimized P2P overlay to minimize server bandwidth cost. Compared to it, the goal of LiveSky is to find an operation guideline, thus, users observe good performance and at the same time the operating costs of the CDN are not too high, but the P2P overlay is unnecessary to be optimal in LiveSky.

the system performance unless seeking the optimal tree [Small et al. 2007]. Consequently, we model the delay between the source and a user in terms of the *level* of that user in the tree.

—Secondly, we imagine that the stream is divided into multiple equal length segments. Each segment contains a few seconds frames. We consider the segment as a basic unit of operation. All clients at the same level in the P2P tree receive a specific segment at the same time.

—Thirdly, we assume that the total upload bandwidth of clients in level k of the P2P tree is always larger than the download bandwidth requirement of clients in level $k+1$. This means that clients in this analytical model never need to *rebuffer*. In other words, they do not experience buffer underflows, where the media buffer does not have any content for the higher-layer application to render.

—Finally, we only consider aggregate measures (i.e., population and time averages) to model the end user properties such as upload capacities and churn rates.

Since our goal is to provide an approximate guideline for operation, we believe that these simplifying assumptions are reasonable. In the following discussion, we adopt and extend the analysis framework of Xu et al. [2006].[4]

Analysis with Node Churn. Suppose the edge SN has sufficient capacity to serve only N_c concurrent streaming sessions. Let P_0 be the total number of clients assigned to this SN that wish to receive this live video stream. Let ρ denote the average fraction of full video bitrate that each supplying peer contributes during a session. Let $N(k)$ represent the total capacity of peers in level k. Note that as a specific case, $N(0) = N_c$. Since we have a total capacity of $N(k-1)$ at level $k-1$ and peers can support ρ sessions on average. We model the node churn using two parameters: the average leave rate σ and the average join rate λ. σ and λ are expressed as a fraction of the total number of current peers. We have

$$N(k) = N(k-1) \times (1-\sigma) \times \rho.$$

Let $P(k)$ be the number of clients in levels $(k+1) \ldots K_0$. Then,

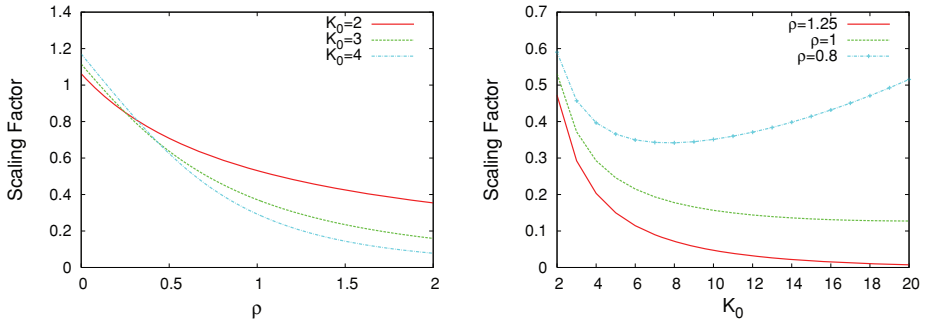
$$P(k) = P(k-1) - N(k-1) - P(k-1) \times \sigma + P_0(1+\lambda-\sigma)^{k-1}\lambda.$$

Let K_0 denote the maximum level in the P2P overlay; i.e., a parameter to capture the maximum acceptable source-to-end delay. Then, by definition $P(K_0) = 0$; that is, it is not acceptable to have clients more than K_0 levels away from the CDN node. we have,

$$S = \frac{N_c}{P_0} = \begin{cases} \frac{(1-\rho)(1+\lambda-\sigma)^{K_0}}{(1-\rho^{K_0})(1-\sigma)^{K_0-1}} & \text{if } \rho \neq 1 \\ \frac{(1+\lambda-\sigma)^{K_0}}{K_0(1-\sigma)^{K_0-1}} & \text{if } \rho = 1. \end{cases} \quad (1)$$

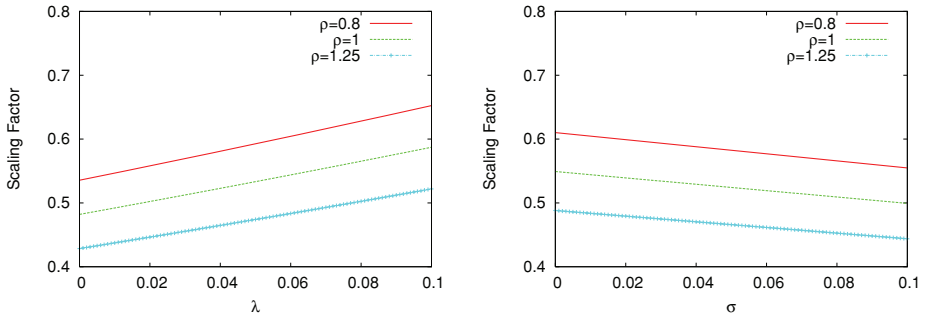
Examples for illustration. Figures 2(a) and 2(b) show how the scaling factor S varies as a function of ρ and K_0 respectively based on the analysis above. We use a constant peer arrival rate $\lambda = 3.6\%$ and a constant peer leave rate $\sigma = 2.4\%$. As expected, when the available P2P bandwidth ρ increases, the system scaling improves and a smaller fraction of the users need to be served by the CDN in Figure 2(a). When $\rho > 1$, by relaxing the delay constraint, that is, increasing K_0, we can reduce the CDN contribution. When $\rho = 1$, the system cannot be scaled arbitrarily, and the scale levels off as a function of K_0. For $\rho < 1$, as the delay bound K_0 is increased, the CDN contribution initially decreases,

[4]There are a few differences between the analysis of Xu et al. [2006] and that presented here. First, we explicitly model the upload bandwidth restrictions on peers and the download streaming requirement at each level. Second, our model of node joins/leaves is more suited to our operational environment. In their framework, peers leave the P2P system once they have contributed some threshold capacity; in our case, peers leave at random.



(a) Vs. ρ (b) Vs. K_0

Fig. 2. Relationship between the scaling factor S and ρ and K_0.



(a) Vs. join rate (b) Vs. leave rate

Fig. 3. Relationship between the scaling factor S and λ and σ.

but subsequently increases beyond a certain critical value. Figures 3(a) and 3(b) show how S varies as a function of λ and σ. As expected, when the join rate λ increases, the system scaling degrades since more server capacity has to be provided to satisfy the requirement of coming clients. Contrarily, when the leave rate σ increases, the system scaling improves since most of the leaving clients are leaf nodes, thus the tree can be rearranged so that the number of end users CDN directly served is reduced. However, CDN do not release the capacities quickly when P2P clients leave system in practice, thus the improvement of scaling is indistinct.

Using the analysis to guide system operation. We use this analysis in conjunction with experiences gained from trial deployments to derive configuration parameters to aid the decision process of each SN. The key is to control N_c, the number of nodes directly served by the edge node, depending on the current load P_0. Each edge SN is *pre-configured* with the decision logic to determine N_c depending on P_0. We briefly describe the decision logic below.

Suppose, we have estimates of ρ, λ, and σ. These can be obtained from understanding user behaviors and available bandwidth characteristics from earlier test deployments and other studies. We also choose a suitable value of K_0, the maximum number of levels in the client-side P2P based on an upper bound on the acceptable source-to-user latency and also on observed performance in field trials. In our deployment, we use $K_0 = 2$. As shown in Figure 4, each edge SN can be in four stages (where N_c^1 is the bound that clients can turn to P2P mode when the total number of users is larger than N_c^1, and N_c^* is simply the total bandwidth capacity of the edge SN divided by the media rate).

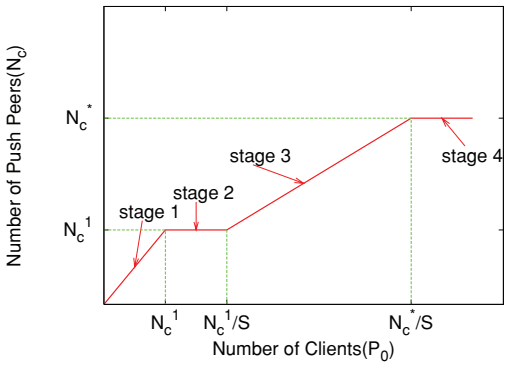


Fig. 4. Different stages in the operation of an edge SN as a function of the number of clients.

—Stage 1, $P_0 \leq N_c^1$. All clients retrieve the content directly from edge server. In stage 1, since the number of end user is low, it is difficult for a client to find other available peers to obtain data, thus, all the clients are categorized as server-pushed clients. Moreover, P2P based system can efficiently distribute hotspot programs but is inefficient to delivery programs with a few viewers so that the content in such programs are directly offered by servers, that is, working in client-server approach.

—Stage 2, $N_c^1 \leq P_0 \leq \frac{N_c^1}{S}$. New clients are served by one of the first N_c^1 peers, that is, we switch from the $K_0 = 1$ mode to the $K_0 = 2$ mode.

—Stage 3, $\frac{N_c^1}{S} \leq P_0 \leq \frac{N_c^*}{S}$. Half the new clients are served directly, rest are redirected to peers.

—Stage 4, $P_0 \geq \frac{N_c^*}{S}$. The edge SN hits its capacity limit N_c^*. New clients have to be redirected to existing peers. Since $K_0 = 2$, we have $S \approx 0.5$ this means that when $P_0 \geq 2 \times N_c^*$, new clients are redirected to other less loaded edge SNs by CDN's GSLB technology.

The specific thresholds (N_c^1 and $\frac{N_c^1}{S}$) when the edge SN transitions between stages are based on the bounds suggested by (1). Since the bound does not account for extra load due to rebuffering events, we make sure that the actual values of N_c^1, $\frac{N_c^1}{S}$ are slightly higher than these analytical bounds. While our current deployment uses $K_0 = 2$, we can extend our system to use larger K_0 values as workloads become even larger and client-side P2P technologies evolve over time.

3.2 Optimizations in Client Side Overlay

Fast startup. Many P2P streaming systems take at least 30 seconds before the video playback starts [Li et al. 2008]. LiveSky reduces the startup delay by two ways. First, we cut the client buffer size to 15 seconds. The smaller size buffer does not only reduce the startup delay, but also reduce the source-to-end delay, while this is impossible in traditional P2P systems that usually adopt larger client buffer. Second, LiveSky leverages the edge SNs to provide fast startup. When a client first sends a request to a edge SN, the SN responds immediately reply it with a pre-specified number of video segments. As the client buffer is filled with the content, it starts to play the video. At the same time, it joins the P2P overlay. Once this is done, it no longer needs the data coming directly from the SN. However, the SNs can not provide fast startup when working in overloaded situation. Thus, the GSLB system commonly redirects a new client to a lightly loaded SN.

Stability under churn. Node churn can cause intermittent disruptions in data delivery. The joining and leaving of clients often brings the change of P2P overlay and the break of certain data links.

Although new data links will be generated eventually, the unstableness of data transmission usually causes the degradation of performance. This can cause frequent interruptions, that is, the media player starts to *rebuffer* because it is waiting for video segments to render, and affect the user viewing experience. However, since LiveSky is a hybrid CDN-P2P system, it can leverage the edge SNs to minimize the impact of such disruptions without requiring heavy-weight mechanism or using large buffers. The key idea is that peers who suffer performance degradation due to churn (e.g., a parent leaves or the connection is poor) can retrieve data directly from the edge SN until the overlay adapts to the churn and these peers can find suitable parent nodes. This ensures continuous video playback with minimal overhead.

Transmission locality. Each edge SN keeps track of clients currently assigned to it (and none others). Each client learns about other peers assigned to its designated edge SN. Since the CDN redirection already takes into account the client and SN locations when assigning a client to an edge SN, this automatically ensures that clients mostly peer with other clients in the same region. This can be viewed as a more direct implementation of a recent proposal for localizing P2P transfers [Choffnes and Bustamante 2008].[5]

NAT handling. A host behind a NAT can receive data but often cannot serve other hosts. Thus, NAT-ed hosts may not contribute any capacity to the system. For example, measurements from CoolStreaming [Li et al. 2007] show that users behind NATs contribute less than one-sixth of the stream rate. This is a serious concern—in many real deployments, a significant fraction of the hosts are behind NATs. Measurements from PPLive show that 60%-80% of hosts in China are behind some kinds of NAT [Huang et al. 2008]. Our measurements (Section 4.5) reveal that more than 90% of the peers in LiveSky are behind NATs.[6] Thus, it is necessary to incorporate NAT traversal mechanisms for the system to scale to a large user population. We adopt well-known techniques such as STUN [Rosenberg et al. 2003] to deal with such connectivity restrictions. In doing so, we leverage the edge SN to serve as an intermediary to facilitate NAT traversal. STUN uses UDP to achieve efficient transmission between clients that behind NATs. We add application-layer ACKs to ensure reliable data delivery over UDP. We also adopt upload bandwidth restriction mechanisms to balance the upload contribution across hosts. For example, in our implementation, we limit the upload bandwidth of each peer to be 150% of the stream bitrate.

4. PERFORMANCE EVALUATION

In this section, we present measurements and analysis from real-world deployments. Several measurement studies have been carried out to analyze pure CDN or pure P2P systems for live and Video-on-Demand (VoD) services [Ali et al. 2006; Hei et al. 2007; Huang et al. 2007; 2008]. However, there is little understanding of how a hybrid CDN-P2P performs in the wild. We believe that this is the first analysis of a commercially deployed hybrid CDN-P2P system for live streaming at a large scale. Previous measurement studies have provided valuable insights into network properties, user behavior patterns, and system dynamics to guide the future development of streaming systems. We hope that our measurement study will also serve a similar role.

[5]Ono [Choffnes and Bustamante 2008] indirectly infers locality information based on "black-box" observations of CDN redirections. Since we already have this information, our approach is more direct. Alternatively, we can use P4P [Xie et al. 2008], Oracle [Thorup and Zwick 2005], or network coordinates (e.g., Dabek et al. [2004]) for localizing P2P transfers; we leave this for future work.

[6]LiveSky only served the users in China and PPLive is mainly used in China.

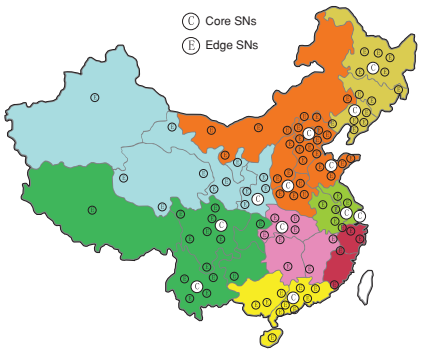


Fig. 5. Deployment of the LiveSky SNs in China for the 17th CPC National Congress.

4.1 Deployment

LiveSky has successfully served several large-scale live streaming events. In this study, we evaluate the system through the trace obtained from one of the largest events—the 17th CPC National Congress[7] on Oct 22, 2007. During this event, ChinaCache deployed about 500 servers in 8 districts in China. There were 50 core SNs distributed throughout the districts. Around the core SNs, there were over 400 edge SNs, scattered within each district. (see Figure 5)

During this event, 400 kbps video stream was provided. In the following discussions, "the users" referred to all of the users who have installed the LiveSky client software, if there is no specific declaration. The peak number of clients was more than 145,000 with an aggregate bandwidth contribution of roughly 34 Gbps (58.6% of the users) from the CDN servers and roughly 17 Gbps from the P2P clients. LiveSky successfully saved more than 40% of the CDN bandwidth cost by integrating P2P technologies. This event exhibited a flash-crowd-like effect within a few minutes of the beginning of the stream. The average viewing time in the event is 13m 28s. Since the event itself only spanned a few minutes, this means that many users stayed and watched a significant portion of the stream. This also suggests that LiveSky was able to provide good user performance and effectively scale the system capacity even with the sudden surge in demand during the flash crowd.[8]

In deployed system, each edge SN operates in isolation and clients in different regions are naturally redirected to local edge SNs. Thus, we can effectively study the system performance and dynamics from the perspective of promise each SN in isolation. Due to skew in the population and viewership, edge SNs see varying request workloads. Some operate in pure CDN mode (i.e., Stage 1) while others switch to the mixed CDN-P2P operation (i.e., Stages 2–4). For the following analysis, we focus on the interesting case: a typical "hotspot" edge region in Beijing that received a sufficiently large request load to cause it to transition into different operating modes. In fact, over 60% of the users in this region were served from the P2P component of LiveSky, compared to the global average of 40%.

[7]The 17th CPC National Congress is a high impact event of tremendous national interest within China comparable to the U.S. Presidential Inauguration. http://www.chinaview.cn/17thcpc/.

[8]Servers are the solution for flash crowd, nevertheless it is nontrivial to properly provision server resource to handle the flash crowd. The reason being the degree of flash crowd depends on a number of dynamic factors including the peer joining pattern during flash crowd, the network dynamics, etc. [Liu et al. 2009]. In an ideal scenario, a server can handle certain number of peers with a given bitrate streamed video, but a hybrid system with peer-assistance can scale to 3–5 times more peers.

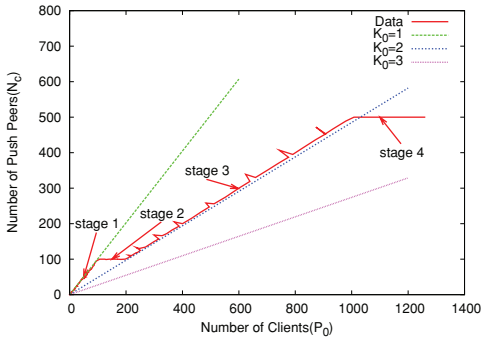


Fig. 6. Adaptive scale control at a busy SN.

4.2 Adaptive Scaling

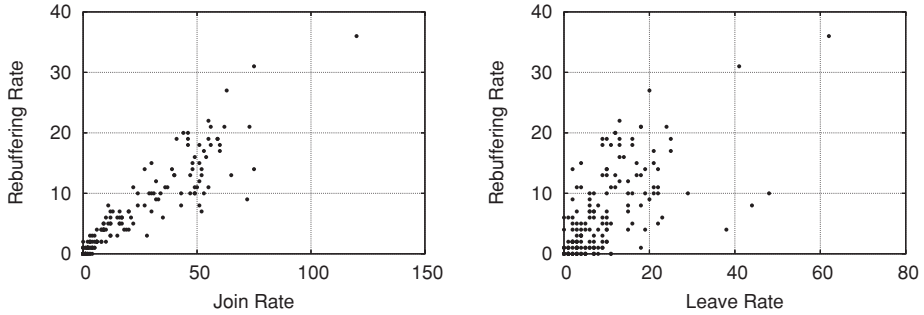
We demonstrate how the adaptive scaling works in practice at the Beijing hotspot (see Figure 6). As discussed in Section 3.1, the key parameter is N_c, the number of *push* peers served directly by the SN. The adaptive control decides a suitable value depending on the offered load (P_0, the x-axis). Here, we configured $N_c^1 = 100$. The bandwidth capacity of the SN is 200 Mbps, which at a media bitrate of 400 kbps translates into $N_c^* = 500$. For reference, we show the expected N_c curves for different values of K_0. We see that the scale control closely follows the expected guidelines, which restrict the minimal N_c suggested by the analysis under certain working environment through estimating of $\rho = 1$, $\lambda = 3.6\%$, and $\sigma = 2.4\%$. These values are obtained from understanding user behaviors and available bandwidth characteristics from earlier test deployments.

4.3 User Experience

Rebuffering Dynamics. Previous work on P2P streaming [Li et al. 2007] primarily captures user experience through the failure rate, where a failure event is defined as the inability of the user to start playing the video. Such a coarse-grained measure only captures the user experience during joins; it does not capture the quality when the user is viewing the event. We define the *rebuffering rate* as the number of clients that rebuffered for playback per minute. We believe that this metric more accurately captures the quality of the user's viewing experience compared to metrics proposed in previous work.

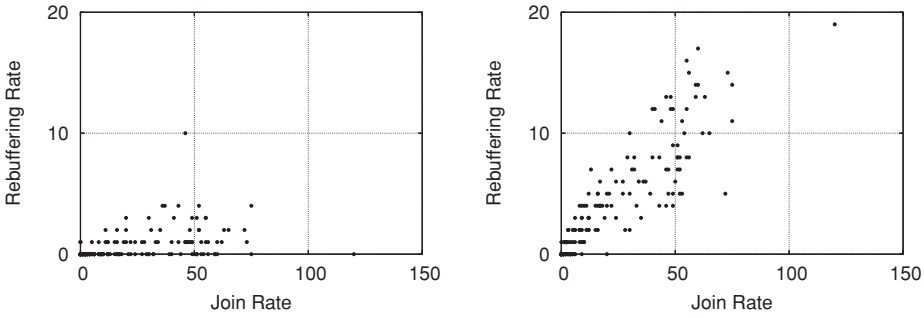
Our result (not shown) has indicated that there is no distinct correlation between rebuffering rate and total users, which is similar to previous work [Li et al. 2007]. Thus, we correlate the rebuffering rate with the system churn—the join and leave rate in a specific interval (i.e., 1 minute), during the live streaming event from 10:00am to 13:00 on Oct 22, 2007. Meanwhile, the factors that join rate and leave rate (only used in this sub-section) are respectively defined as the number of clients that join the system and leave the system per minute. Figure 7(a) and Figure 7(b) respectively show the rebuffering rate against the join rate and leave rate in each specific minute interval during the event. While the results denote that there is a reasonably strong correlation between the rebuffering rate and join rate, but little correlation with the remaining factor.

Given that the join rate is the dominant factor affecting the rebuffering rate, we do a more fine-grained analysis of the relationship between the rebuffering rate and the join rate. For this, we classify clients along two dimensions: (1) how long they have been in the system (e.g., old vs. new, that whether staying in the system more than one minute), and (2) whether they are being served by the CDN node



(a) Vs. Join Rate (b) Vs. Leave Rate

Fig. 7. Rebuffering rate vs. system churn.

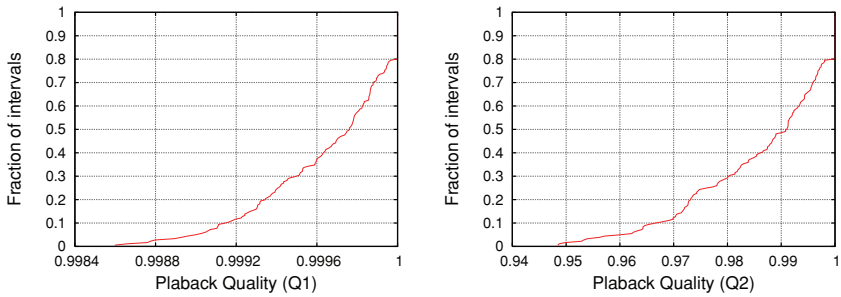


(a) Old P2P clients (b) New P2P clients

Fig. 8. Correlation between join rate and rebuffering rate for different classes of clients.

or by the P2P system (Push vs. P2P). Since the *Push* clients (no matter old or new) directly gain the data from the CDN servers that is more stable, it is inevitable that there is little or no correlation between the rebuffering rate of *Push* clients and the join rate. Our results also indicate this to be true (not shown). Figure 8 correlates the rebuffering rate for *P2P* clients to the aggregate join rate. There is little correlation between the rebuffering rate of old P2P users and the join rate (Figure 8(a)). The results demonstrate that the churn that new clients joining do not obviously affect the viewing experience of old clients which have stayed in the system for a longer time. There is however, a stronger correlation between the rebuffering rate of new P2P users and the join rate (Figure 8(b)). Even this correlation is substantially less than that observed in P2P-only streaming systems [Li et al. 2007]. This suggests that LiveSky is more effective at insulating users from churn-induced effects compared to pure-P2P systems. Additionally, the results demonstrate that we need further specially improve the performance of new P2P clients because they experience more viewing disruptions.

Aggregate Quality Indices. We define two quality indices to measure the users' viewing quality in a given measurement interval. The first index is $Q_1 = \frac{T_P}{T_P+T_B}$, where T_P is the total (across all clients) time spent in media playback and T_B denotes the total time spent in rebuffering in this measurement interval (i.e. 1 minute). The second index is $Q_2 = 1-\frac{N_B}{N_T}$, where N_T is the total number of clients and N_B is the number of clients that experienced some rebuffering in this measurement interval (i.e. 1 minute). Ideally, we want these two quality indices to be as close to 1 as possible, that is, perfectly smooth playback for all users. Figure 9 shows the distribution of these two playback quality indices over the



(a) Q1: Buffered time (b) Q2: Buffering events

Fig. 9. Metrics to capture playback quality.

entire duration of the live stream. We see that even the worst case values of these quality measures are greater than 0.95 and that the median values for Q_1 and Q_2 are 0.9998 and 0.99 respectively. These show that LiveSky delivers high quality viewing experience.

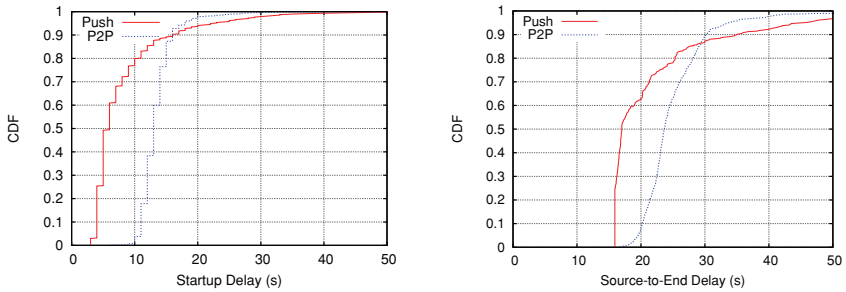
Delay Comparisons. Anecdotal evidence suggests that the startup delay is a crucial factor in user quality—users are likely to get frustrated and leave if they perceive high startup delays [Liu et al. 2008]. In LiveSky, more than 85% of the clients wait less than 15s for playback to commence from clicking on the hyperlink. Measurements from P2P streaming systems like CoolStreaming show that most clients observe startup delays greater than 30s. Thus, it is clear that the hybrid architecture provides significantly faster startup performance. In addition, Figure 10(a) illustrates the comparison of startup delay between push clients and P2P clients. It shows that most of the push clients wait shorter time when startup (e.g., 80% of the push clients wait less than 10s, compared to 80% of the P2P clients wait between 10s and 15s). However, there are some users that waiting more than 15s (even 30s) to start playback including both the P2P clients and Push clients. Since the links between the edge SN and some push clients are unsatisfactory, at the same time the push clients only retrieve missing data from edge SN, some push clients have to experience longer startup delay.

The source-to-end delay is another crucial factor in user quality, which is defined as the latency a video segment has been sent out from the server until it has been viewed on client side. In LiveSky, more than 85% of the clients experience less than 30s source-to-end delay. It is clear that the hybrid architecture provides significantly better source-to-end delay than traditional P2P streaming systems that adopted very large client buffers. Figure 10(b) illustrates the comparison of source-to-end delay between push clients and P2P clients. Most of the Push clients experience better quality of source-to-end delay than P2P clients.[9]

4.4 Network Friendliness

We can evaluate the locality of the P2P transmissions in LiveSky by analyzing the effectiveness of the DNS-based redirections used by the CDN as our P2P overlay is localized on a per-SN basis. For example, if the CDN redirection was accurate in assigning 90% of the clients located in Beijing to the edge SN located in Beijing, then most of the P2P transmissions associated with this edge SN will be localized. The CDN redirection maps the IP address of the local DNS server of the client to a geographic location using a IP to location database and redirects the client to a suitable edge SN. There are two

[9]We compare the user experience between push clients and P2P clients as the comparison between pure CDN and hybrid system. The results indicate that push clients get better quality than P2P clients, however, P2P clients also obtain good enough experience compared to pure P2P systems.



(a) Startup Delay (b) Source-to-End Delay

Fig. 10. Delay Comparison between Push Clients and P2P Clients.

potential sources of error in this redirection: 1) clients may have misconfigured their local DNS server (e.g., using a static DNS server instead of using a DNS server provided by the upstream ISP), and 2) the IP to location mapping might be out of date. We analyze the accuracy of the redirection by checking if a client's public IP address maps to the same region as the edge SN assigned to it by the GSLB. In more than 80% of the cases the client is assigned to an edge SN in the same region. Thus, most of the P2P traffic in LiveSky is localized within the region and this ensures a "network-friendly" operation.

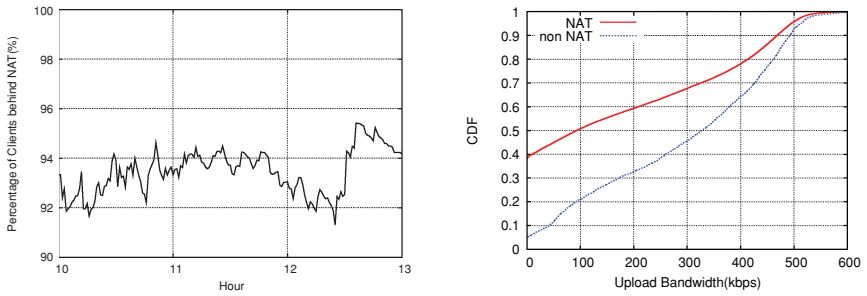
4.5 Upload Contribution

In the system, each client reports its private IP address (e.g., 10.9.*) to the edge server and the edge server can obtain its public IP address from the received packets, thus, we can determine if this client is behind a NAT. During the Chinese Congress event, more than 90% of end hosts are behind NATs (Figure 11(a), which indicates that the percentage of NAT-ed hosts is always larger than 90% in each 1 minute interval during the whole event). Figure 11(b) shows the distribution of upload bandwidth contribution of both NAT-ed and non-NAT hosts. The CDF ends at 600kbps only because we manually limited the upload contribution to be 1.5× the video bitrate ($bitrate = 400kbps$) in the client software. While 37% clients behind NAT could not provide upload capacity[10] (larger than 5% of non-NAT clients that do not offer upload capacity), the majority of NAT-ed hosts contribute significantly. For example, nearly 30% of NAT-ed hosts contribute 1× the video bitrate. Moreover, without considering the clients that do not provide their bandwidth capacity, the distribution of upload contribution is similar between remaining NAT-ed and non-NAT hosts. Compared to measurements from CoolStreaming systems [Li et al. 2007], this result suggests a greater contribution from NAT-ed hosts.

5. ENHANCEMENTS AND FUTURE DIRECTIONS

In this section, we discuss several recent improvements of LiveSky to overcome some limitations. In addition, we highlight some future directions.

[10]Such result may caused by the clients behind *Symmetric NAT* which are hard to traverse in *LiveSky*, while a previous study [Huang et al. 2008] has denoted that about 30% of the NAT types is *Symmetric NAT* in China. We will confirm it and improve our system in future work.



(a) Percentage of NAT-ed hosts (b) Contribution of NAT-ed and non-NAT

Fig. 11. Percentage and Contribution of NAT-ed and non-NAT hosts.

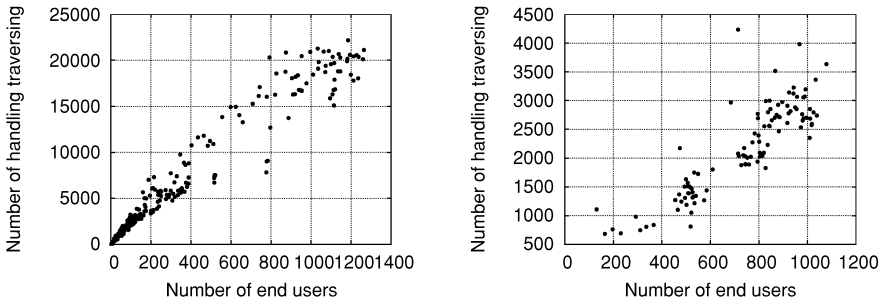
5.1 System Enhancements

P2P Group Handling. As described in Section 2 and Section 3, the P2P overlays are localized on a per-edge SN basis. Such mechanism has some limitations. First, it is hard to handle the SN node or network failures. Though the GSLB system can redirect the client requests to a lightly loaded SN and monitor the SNs to detect the SN node or network failures, it could not react to failures rapidly since the redirection is DNS-based. Second, the client geographical distribution is skew. For example, our analysis from several large scale live events have exhibited the situation that the number of end users from Beijing, Shanghai, Guangdong, etc., is largely greater than the numbers in other regions in China Mainland. As a result, some regions will become "hotspot" and the SNs served for these regions are easily overloaded. However, some other SNs with fewer end users may only operate in pure CDN mode, which is cost-ineffective.

To overcome these limitations, we enable the peers to communicate with other clients assigned to different edge SNs. In the design of per-edge SN basis P2P overlays, each edge SN only maintains the information of the clients that assigned to it. Thus, the clients only obtain the client information in the same edge region from edge SN and establish communication with them. Recently, we enable the edge SN not only maintains the clients assigned to it but also the client information assigned to other edge SNs by exchanging the client information between edge SNs periodically. Thus, clients obtain the edge SNs and clients information of other edge regions from local edge SN. Afterwards, the client can rapidly retrieve data from other edge regions (either from other edge SNs or clients) when detecting local edge SN node or network failures. In practice, the edge SNs are separated into several districts (e.g., according to the 8 districts discussed in Section 4). We only enable the edge SNs in the same district to exchange client information, and disable the exchanging between different districts. Thus, the P2P traffics are localized in each district. At the same time, such district-based overlay organization can efficiently balance the server load among the edge SNs in the same district.

We leave some detailed discussion and evaluation studies for future works, such as, 1) how to determine the different districts with simultaneous considering of the performance, network-friendliness, and cost, 2) how to effectively deploy the servers in practice, and 3) how to integrate existing transmission locality approaches (e.g., P4P, Oracle, or network coordinates).

NAT Traversing. Our evaluation results have indicated that our NAT traversing mechanism is efficient as many NAT-ed hosts have contributed upload bandwidth. Because we leverage the edge SN to serve as an intermediary to facilitate all the NAT traversal, the edge SN may become very heavily. Our evaluation results have denoted that the edge SN handled numerous NAT traversing requests per



(a) 17th CPC Progress (b) Recent Event in Feb 2010

Fig. 12. Comparison of NAT traversing handling on edge SN between two events.

minute (e.g., even 10-20 times the total number of end users) during the event (see Figure 12(a)). To reduce the stress on edge SN, we leverage the client (e.g., client A) as an intermediary to facilitate the NAT traversing for other peers (e.g., client B and C) when both B and C are the neighbors of A. Edge SN handle the traversing request after the NAT traversing among clients is failed. Recent deployment after the enhancement has denoted that the number of handling NAT traversing requests on edge SN has been efficiently reduced (e.g., less than 5$\times$ the total number of end users per minute, as shown in Figure 12(b)).

5.2 Future Directions

Firstly, improving P2P technologies for live streaming and VoD like applications is still an ongoing area of research. LiveSky can leverage developments in this area for providing even better performance at scale. We would like to achieve more fine-grained instrumentation of clients to be able to better diagnose the root causes of poor client performance (e.g., playback disruptions and slow startup delay). We are considering using scaling option of video coding for fast startup in our system, thus, end users can start to view the video more rapidly though the users may experience several seconds low-quality video stream. Secondly, LiveSky is deployed on the CDN Infrastructure, which is responsible for supporting multiple applications (e.g. Live, VoD, and file downloading, etc.) and multiple channels (e.g., different channels that are both live or VoD). We are currently focusing on the research on how to efficiently allocation the resources among these applications or channels, especially during LiveSky is used to broadcast large scale live event. Finally, we plan to incorporate better security features e.g., to only allow authorized users to access the stream, ensuring that peers do not intentionally deny service etc.

6. CONCLUSIONS

There has been tremendous interest in both academic and industrial research communities on combining the benefits of traditional CDN architectures and P2P systems for live video streaming. However, there have been few reported works of real deployments that validate the promise of such a hybrid CDN-P2P architecture. Consequently, we do not have a real understanding of how such an architecture performs “in the wild.” The contribution of this article is a practical one—it presents our experiences in designing and deploying a real-world hybrid CDN-P2P system for live streaming called LiveSky that helps bridge this gap.

In designing LiveSky, we use existing P2P technologies and integrate them with minimal changes to the existing CDN infrastructure. In doing so, we make specific design choices to ensure that the system scales with the number of users, at the same time provides the users with a good viewing

experience, that is, low start-up delay and minimal disruptions due to rebuffering, even under churn. We leverage the presence of the infrastructure nodes and the CDN redirection mechanisms to address some well-known deficiencies in P2P systems such as enabling users behind NATs to contribute upload bandwidth and localizing P2P traffic to enhance the network friendliness. From the evaluation results, we can see that LiveSky system can make a good trade-off among the properties of scalability, QoS guarantee, and availability. The outcomes of this study can bring valuable insights for the future improvement and development of live streaming technologies and applications.

REFERENCES

ALI, S., MATHUR, A., AND ZHANG, H. 2006. Measurement of commercial peer-to-peer live video streaming. In *Proceedings of the International Workshop on Recent Advances in P2P Streaming*.

CASTRO, M., DRUSCHEL, P., KERMARREC, A. M., NANDI, A., ROWSTRON, A., AND SINGH, A. 2003. SplitStream: High-bandwidth content distribution in cooperative environments. In *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*. 298–313.

CHOFFNES, D. R. AND BUSTAMANTE, F. E. 2008. Taming the torrent: A practical approach to reducing cross-isp traffic in peer-to-peer systems. In *Proceedings of ACM SIGCOMM Data Communication Festival (SIGCOMM)*. 363–374.

CHU, Y., RAO, S. G., SESHAN, S., AND ZHANG, H. 2000. A case for end system multicast. In *Proceedings of ACM SIGMETRICS*. 1–12.

DABEK, F., COX, R., KAASHOEK, F., AND MORRIS, R. 2004. Vivaldi: A decentralized network coordinate system. In *Proceedings of ACM SIGCOMM Data Communication Festival (SIGCOMM)*. 15–26.

DARLAGIANNIS, V., MAUTHE, A., AND STEINMETZ, R. 2007. Sampling cluster endurance for peer-to-peer based content distribution networks. *Multimedia Syst. 13*, 1, 19–33.

DISCHINGER, M., MISLOVE, A., HAEBERLEN, A., AND GUMMADI, K. P. 2008. Detecting BitTorrent blocking. In *Proceedings of ACM SIGCOMM Internet Measurement Conference (SIGCOMM IMC)*. 3–8.

GANNES, L. 2009. The Obama Inauguration Live Stream Stats. `http://newteevee.com/2009/01/20/the-obama-inauguration-live-stream-stats/`.

HEI, X., LIANG, C., LIANG, J., LIU, Y., AND ROSS, K. W. 2007. A measurement study of a large-scale P2P IPTV system. *IEEE Trans. Multimedia 9*, 8, 1672–1687.

HUANG, C., LI, J., AND ROSS, K. W. 2007. Can Internet video-on-demand be profitable? In *Proceedings of ACM SIGCOMM Data Communications Festival (SIGCOMM)*. 133–144.

HUANG, C., WANG, A., LI, J., AND ROSS, K. W. 2008. Understanding hybrid CDN-P2P: Why Limelight needs its own red swoosh. In *Proceedings of ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*. 75–80.

HUANG, G. 2007. Keynote: Experiences with PPLive. In *Proceedings of ACM SIGCOMM Peer-to-Peer Streaming and IP-TV Workshop (SIGCOMM P2P)*.

HUANG, Y., FU, T. Z. J., CHIU, D.-M., LUI, J. C. S., AND HUANG, C. 2008. Challenges, design and analysis of a large-scale P2P-VoD system. In *Proceedings of ACM SIGCOMM Data Communications Festival (SIGCOMM)*. 375–388.

KARAGIANNIS, T., RODRIGUEZ, P., AND PAPAGIANNAKI, K. 2005. Should Internet service providers fear peer-assisted content distribution. In *Proceedings of ACM SIGCOMM Internet Measurement Conference (SIGCOMM IMC)*. 63–76.

KIRKPATRICK, M. 2008. The numbers are in, live video online is blowing up. `http://www.readwriteweb.com/archives/live_video_big.php`.

LI, B., QU, Y., KEUNG, Y., XIE, S., LIN, C., LIU, J., AND ZHANG, X. 2008. Inside the new coolstreaming: Principles, measurements and performance implications. In *Proceedings of Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. 1031–1039.

LI, B., XIE, S., KEUNG, G. Y., LIU, J., STOICA, I., ZHANG, H., AND ZHANG, X. 2007. An empirical study of the CoolStreaming system. *IEEE J. Select. Areas Comm. 25*, 9, 1627–1639.

LIU, F., LI, B., ZHONG, L., LI, B., AND NIU, D. 2009. How P2P streaming systems scale over time under a flash crowd. In *Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS)*.

LIU, J., RAO, S. G., LI, B., AND ZHANG, H. 2008. Opportunities and challenges of peer-to-peer Internet video broadcast. *Proc. IEEE 96*, 1, 11–24.

PAKKALA, D. AND LATVAKOSKI, J. 2005. Towards a peer-to-peer extended content delivery network. In *Proceedings of the 14th IST Mobile and Wireless Communications Summit*.

RODRIGUEZ, P., TAN, S. M., AND GKANTSIDIS, C. 2006. On the feasibility of commercial, legal P2P content distribution. *ACM SIGCOMM Communications Review (SIGCOMM CCR) 36*, 1, 75–78.

ROSENBERG, J., WEINBERGER, J., HUITEMA, C., AND MAHY, R. 2003. STUN—Simple traversal of user datagram protocol (UDP) through network address translators (NATs). IETF RFC 3489.

SMALL, T., LI, B., AND LIANG, B. 2007. Outreach: Peer-to-peer topology construction towards minimized server bandwidth costs. *IEEE J. Select. Areas Comm. 25*, 1, 35–45.

THORUP, M. AND ZWICK, U. 2005. Approximate distance oracles. *J. ACM 52*, 1, 1–24.

VANCE, A. 2009. News sites struggle to stream Obama video. `http://bits.blogs.nytimes.com/2009/01/20/news-sites-struggle-to-stream-obamas-innaguration-speech/?apage=1`.

VENKATARAMAN, V., YOSHIDA, K., AND FRANCIS, P. 2006. Chunkyspread: Heterogeneous unstructured tree-based peer-to-peer multicast. In *Proceedings of Annual International Conference on Network Protocols (ICNP)*. 2–11.

WANG, F., XIONG, Y., AND LIU, J. 2007. mTreebone: A hybrid tree/mesh overlay for application-layer live video multicast. In *Proceedings of the International Conference on Distributed Computing Systems*. 49–56.

WU, C., LI, B., AND ZHAO, S. 2009. Diagnosing network-wide P2P live streaming inefficiencies. In *Proceedings of Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. 2731–2735.

XIE, H., YANG, Y. R., KRISHNAMURTHY, A., LIU, Y. G., AND SILBERSCHATZ, A. 2008. P4p: Provider portal for applications. In *Proceedings of ACM SIGCOMM Data Communications Festival (SIGCOMM)*. 351–362.

XIE, S., LI, B., KEUNG, G. Y., AND ZHANG, X. 2007. Coolstreaming: Design, theory, and practice. *IEEE Trans. Multimedia 9*, 8, 1661–1671.

XU, D., KULKARNI, S., ROSENBERG, C., AND CHAI, H. 2006. Analysis of a CDN-P2P hybrid architecture for cost-effective streaming media distribution. *Multimedia Syst. 11*, 4, 383–399.

ZHANG, X., LIU, J., LI, B., AND YUM, T. S. P. 2005. CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming. In *Proceedings of Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. 2102–2111.

Received March 2010; revised May 2010; accepted May 2010