Reverse Engineering the Peer to Peer Streaming Media System

Prof. Dhaval Jha[1] Priyanka Kaushik [2] Rohan Jain [3] Sanu Kumar [4]
[1] Professor
[1, 2, 3, 4]Department of Computer Engineering
[1,2,3,4] Nirma Institute of technology, Gujarat, India.

Abstract—Peer to peer (P2P) content distribution network like BitTorrent (BT) is one of most popular Internet applications today. Its success heavily lies on the ability to share the capacity of all the individuals as a whole. This paper focuses on the operation and dynamics of P2P systems .We split our understanding objective into the sub objectives which follows following sequence first we will understand the working principle through the communication protocol crack. Then we comprehend the streaming content-delivery principle and locate the measurable parameters which can be used to evaluate the system performance, understand the P2P network through the models of startup process and user behavior, and analyze the engineering design objectives. For reaching these goals a well-designed process of reverse engineering is used to setup an ease for analysis, results and conclusion.

Keywords: Reverse engineering, P2P network, live peers, VoD peers, seeders, leechers.

I. INTRODUCTION

Peer-to-Peer (P2P) networking has recently emerged as a new paradigm to build distributed network applications. The basic design philosophy of P2P is to encourage users to act as both clients and servers, namely as peers. In a P2P network, a peer not only downloads data from the network, but also uploads the downloaded data to other users in the network. The uploading bandwidth of end users is efficiently utilized to reduce the bandwidth burdens otherwise placed on the servers. P2P file sharing applications, such as [4, 10], have been widely employed to quickly disseminate data files on the Internet. More recently, P2P technology has been employed to provide media streaming services. Several P2P streaming systems have been deployed to provide on demand or live video streaming services over the Internet [6, 32, 25, and 26]. Our recent measurement study [16] of a P2P live video streaming system shows that, in early 2006, more than 200,000 simultaneous users watched the live broadcast of an 4-hour event at bit rates from 400 to 800 kpbs. The aggregate required bandwidth reaches 100 gigabits/sec, while Akamai reportedly has roughly 300 gigabits/sec bandwidth in its entire network at the end of year 2006. There are two existing P2P media streaming system they are P2P live streaming system and P2P video-on-demand media system.

A. *P2P live streaming system*

In a live streaming session, a live video content is disseminated to all users in realtime. The video playbacks on all users are synchronized. To the contrary, video-on-demand users enjoy the flexibility of watching whatever video clips whenever they want. The playbacks of the same video clip on different users are not synchronized.

B. *Video-on-demand media system*

Video-on-demand service (VoD) allows users to watch any point of video at any time. Compared with live streaming, VoD offers more flexibility and convenience to users and truly realizes the goal of watch whatever you want whenever you want. VoD has been identified as the key feature to attract consumers to IPTV service. In VoD service, although a large number of users may be watching the same video, they are asynchronous to each other and different users are watching different portions of the same video at any given moment. Tree-based P2P system is originally designed to function as IP multicast at the application layer without underlying network layer's support.

II. SIGNALING CRACK AND NETWORK MEASUREMENT

Reverse-engineering-based protocol crack is the first step. It helps understand the working mechanism in depth, but also makes our large-scale measuring possible by developing network crawler. To the best of our knowledge, the work presented here and in related papers by the same authors and colleagues is the first in the world who succeeded in cracking and measuring all the top popular P2P streaming media systems in large scale.

A. *Brief description of P2P VoD system*

Referring to Fig.1, a typical P2P media streaming system uses few servers to serve large number of audiences (named as *peer*) with both live and VoD programs (Ali et al., 2006; Heiet al., 2007a; Zhang, et al., 2005). There are significant different design concerns about P2P VoD system and live system: *i).* VoD peer uses much more storage space to cache nearly the whole video in long term than live peer to cache very few latest contents temporarily. Besides, VoD peer may share all the cached contents even if he is in a different channel. (b) P2P live system is of source-driven such that seeder controls the content feeding rate, while P2P VoD system is of receiver-driven and each peer controls playback rate by himself. Unlike live peer, VoD user has more flexibility to choose different playback patterns, such as skipping, fast forwards and fast backwards.

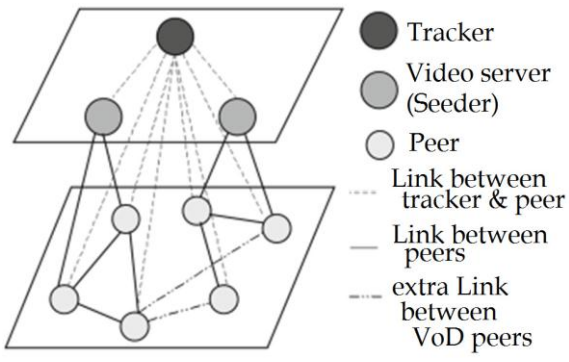

Fig. 1: The system structure

B. The communication protocol cracking

In general, the protocol crack is a cycling procedure including following steps:

1) Network sniffer/measurement:

In the first step, performed using a client sniffer, we capture the interactive packets between the local peer and others. We get to know the important protocol messages must be there such as *shake hand* message, buffer map message (*BM*), and peer list message (*peerlist*), based on existing research reports and our experience. By connecting those types of message to the sniffer trace, it is not difficult to distinguish all kinds of message, even though some messages' functions are unknown.

2) Protocol message guess:

Next, we observe each message in different dimensions, including the dimensions of time, channel and peer. For facilitating observation, we use a small software (developed by us) to extract the wanted messages with some query conditions, such as source IP/port, destination IP/port and message type, from the traces. From the extracted records, we can see many regular patterns which help parse the detailed format of each message. Of course, this way doesn't always work well, for the minority of messages can't be explained. So, we don't neglect any available reference information, e.g., we have ever found the fields of total upload/download count and upload/download speed per peer contained in BM based on the information displayed in PPStream client window. In general, we crack more than 80% messages for PPLive, PPStream and UUSee.

3) Test and Confirmation:

In this stage, we analyze and validate the interactive sequences of messages. We guess and try different interactive sequences until the normal peer or tracker gives the right response. At last, nearly all the guesses are confirmed by our successfully and legally access to the real network.

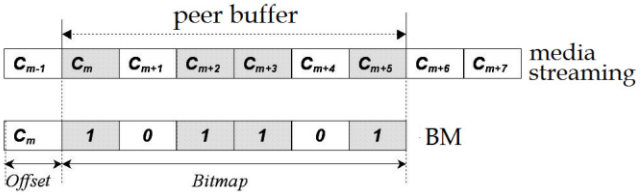

Fig. 2: Buffer and buffer map

III. REVERSE ENGINEERING ANALYSIS FROM A PEER'S VIEWPOINT

Like the BT system, live peer may play roles of *leecher (watcher)* or *seeder*. A seeder has the complete video, while a leecher hasn't. In a P2P live streaming media system, all peers are watchers and a few content servers are seeders. On the other hand, a P2P VoD system also contains two roles. However, they are not classified based on whether a peer has a complete file or not. Although most VoD peers do not own a complete video, he can share it once he is online regardless of the viewing channel. In a channel, we name a peer never downloading from others as a *contributor*, and a peer downloading from others as a *watcher*. VoD *watcher* is just like live watcher in many aspects, while VoD *contributor* may not necessarily have a complete file. As a *contributor*, the VoD peer may upload one movie while watching another. A significant difference of a VoD system from a live system is that *contributors* largely outnumber *watchers*. Our measurement shows that about two-third peers are *contributors*.

A. Live peer behavior in P2P streaming media system

Nearly all existing studies simply assume a stable playback rate. Thus we start with the problem of video playback rate measurement to launch our analysis. Then, we raised the questions of how a peer reaches its stable playback state, and whether and how a peer can keep in good shape.

1) Playback rate and service curve

Intuitively, the forward BM offset with time t in peer p, noted as $fp(t)$, is connected to its playback rate. According to our experience, small rate changes are hidden if we were to visualize $fp(t)$ directly as a time sequence. Instead, a curve of *rt-fp* (t) with proper value of playback rate r can make the changes obviously. However, to check every peer's playback rate is a hard job. In practice, each peer has its own playback rate which roughly equals to the system playback rate, otherwise video continuity cannot be ensured. Thus, a system playback rate should be found as a common reference for observing peer offset progress. We describe the system playback process by a *service curve* $s(t)$. It is reasonable to use the system maximal chunk ID at any time t as $s(t)$, and then playback rate is $r(t) = ds(t)/dt$ For a channel with playback rate variations, the playback rate vs. time should be a piecewise linear function.

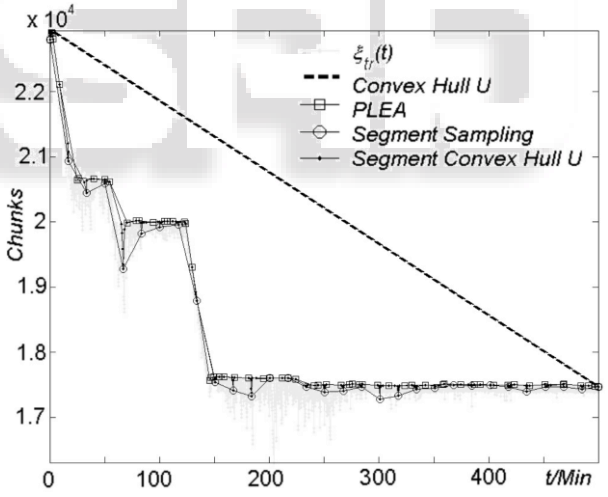

Fig. 3: PLEA vs. others algorithms

The procedure of finding the rate change is similar to the method in estimating the clock skew in network delay measurements. In (Zhang, 2002), people presented "Covex_Hull_L" algorithm and a segmented algorithm, which are denoted as CHU and SCHU respectively in our research, to calculate the network delay. However, referring to Fig.3, the convex envelope (dash line) calculated by CHU fails to reflect the rate changes in medium time scale in our trace 070502. Through slightly modifying SCHU algorithm, we get a new method called Piecewise Line Envelop Approximation (PLEA) (Li & Chen, 2009). The *rate reset time* $\{tk\}$ and *reset rate* $\{rk\}$ is simply the turn point and slope of each segment in the piecewise line calculated by PLEA respectively. The key of PLEA is to take convex hull only in small time scale and follow the rate variation in

medium time scale. Thus, a parameter named as *follow-up time* Δ is introduced. An observed point will be kept if the time difference between this point and previously saved point is larger than Δ Unlike SCHU, our segmentation is automatically adjusted during the calculation procedure without reassigned or fixed. The square marked line in Fig.3 shows the result of PLEA with Δ=1500s. It fits the envelope of trace quite well. Comparing PLEA to SCHU in Fig.3, the result of PLEA is much smoother.

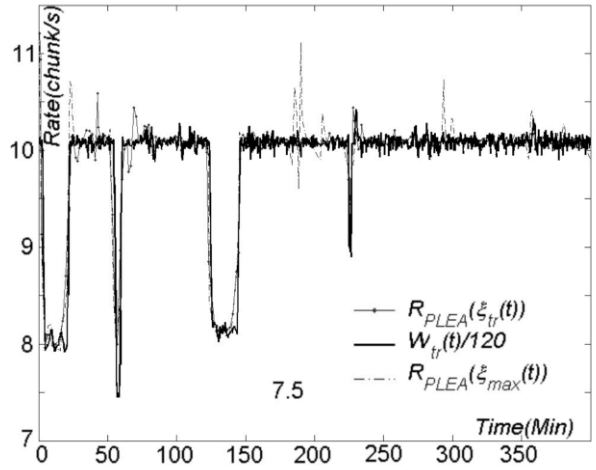

Fig. 4: Comparison of our algorithms

Besides PLEA, we have an occasional but very useful finding during reverse engineering. In PPLive, the seeder's buffer width *Wtr*(*t*) reported by tracker, which is the difference of seeder's scope minus its offset, is always equals to the product of 120 and current playback rate *r*, i.e., *Wtr*(*t*)=120*r*(*t*). For validation, we draw all the rate curves calculated from PLEA of tracker scope ξ*tr*(*t*) and peers max scope ξ*max(t), i.e., RPLEA*(ξ*tr*(*t*)) and *RPLEA*(ξ*max(t)*), as well as *Wtr*(*t*)/120 in the same trace in Fig.4. All rate curves match well except some individual points. Thus we have following observations: For any PPLive channel, the instantaneous rates deduced from both tracker scope and peer maximal scope equal each other, and they are about 1/120 of the seeder's *buffer width*, i.e., *RPLEA*(ξ*tr*(*t*))=*RPLEA*(ξ*max*(*t*))=*Wtr(t)*/120. Then new questions are naturally raised. Whether the system has took the rate variations into account in design? When rate change occurs, can that lead a peer to restart? All such questions involve a primary problem, what is operating mechanism of a peer, especially in its early stage.

2) Model-based observation of peer initial offset selection

We name the peer's first wanted chunk as the *initial offset* θ. *We* reconstruct a peer startup model in Fig.7 to explain the importance of initial offset. Assuming a constant playback rate *r*, *service curve s (t)* is a global reference. Assuming a constant seeder buffer width *Wtk*, we have the seeder's offset curve *ftk* (*t*) =*s* (*t*)-*Wtk* below *s (t)*. The host's first neighbor *p*'s *offset curve* and *scope curve* (of its largest chunk ID) are *fp (t)* and ξ*p (t)* respectively. Since the number of successive chunks in a buffer indicates how long the video can be played continually, we follow (Hei et al., 2007b) to name that as the *buffer's playable video Vp(t)*, correspondingly the *peer's playable video vp(t)=fp(t)+ Vp(t)*, which is also drawn in Fig.5. The initial offset is very important for that, once it, saying θ*h*, is chosen at certain time *th*, the host's offset lag *Lh=s*(*t*)-*fh*(*t*) is totally determined. As shown in Fig.7, *fh* (*t*) begins to increase after the τ_S; meanwhile *s* (*t*) has increased $r\ \tau_S$. Since the host initial offset lag is $L\theta=s(th) - \theta h$, its *offset lag* at last is $Lh=L\theta+r\tau_S$. Lh is the playback lag, but also the possible maximum buffer width. It means θ*h* can affect the system sharing environment.

For simplicity, we assume the *initial offset* decision is based on the host's *first neighbor p*. Then, host *h* faces two alternatives -- based on either the tracker or its first neighbor. Seeing Fig.7, at time *th*, host *h* gets values of *s*(*th*)=*TkOffMax* and *ftk*(*th*)= *TkOffMin* from tracker, and values of ξ*p*(*th*), *vp*(*th*) and *fp*(*th*) from its first neighbor *p*. Then the host should choice its θh between *fp*(*th*) and $\xi p(th)$, beyond which scope no chunk is available. For further explanation, the chunk □*h* will shift out of the neighbor *p*'s buffer at time *th*+($\theta h - fp(th)$)/*r*. Large θ*h* lets host *h* have more time to fetch this chunk. However, as too large θh will lead to a very small *offset lag*, host's *buffer width* maybe not large enough for a good playback performance. So what are the design principles behind the initial offset selection? We extract the marked points shown in Fig.7 at time *th* from our 2502 experiments, and draw them as a function of sorted experiment sequence in ascending order of *Wtk* and *Wp(t)* in Fig.8 where we take *ftk*(*th*) as the horizontal zero reference. The red lines are the seeder's buffer width $\pm Wtk=\pm(s(th)-ftk(th))$. The top one is *Wtk* and the bottom one is −*Wtk*. Clearly, PPLive mainly serves two playback rates: 10 chunks/s on the right area and 6 chunks/s on the left area. The *black '.'* and *green 'x'* stand for $\xi p - ftk$ and $fp - ftk$ respectively, the distance between which marks in each experiment is peer p's buffer width $Wp=\xi p - fp$. Similarly, the vertical distance between top red '-' and *green 'x'* is peer p's offset lag $Lp=s-fp$. Thus, Fig.8 confirms that PPLive takes certain variable buffer width scheme. Furthermore, seeder has a larger buffer than normal peer. The *blue '*'* is hosts relative initial offset lag $\theta h - ftk$. Obviously, PPLive doesn't adopt a fixed initial offset lag scheme, or else all *blue '*'* would keep flat. Actually the *blue '□'* and *green 'x'* have a similar shape, which means that *initial offset* may adapt to *first neighbor p's* buffer condition.

We think certain kind of *Proportional Placement* (PP) strategy (Li & Chen, 2008a) can be introduced to make the decision of *initial offset*. Referring to Fig.8, the distance of the initial offset to its first received BM's offset is somehow proportional to the first neighbor's buffer width $Wp=\xi p$ or the first neighbor's offset lag $Lp=s-fp$. Thus, we guess PPLive chooses the *initial offset* either by $\theta h=fp+\alpha_W Wp$ or $\theta h=fp+\alpha_L Lp$, where the α_W and α_L are the scale coefficients. Based our measurement, both PDFs of $\alpha_W=(\theta h-fp)/Wp$ and $\alpha_L=(\theta h-fp)/Lp$ have the very high peaks at the same coefficient 0.34. The scaled errors of $100(\alpha_W-0.34)$ is shown with the cyan color in Fig.8. It seems that PPLive more likely uses a scheme based on the first neighbor's buffer width since α_W has a more sharp distribution. To check whether the selected *initial offset* θ is easy to download, as well as to evaluate whether PPLive has been designed to make host set its

initial offset at the most suitable point locally or globally, we have studied the chunk availability. As a result, a host usually receives BMs from 4.69 peers before fetching any chunks. In more than 70% experiments, host can fetch chunks around θ from at least 3 neighbors. It indicates a good initial downloading performance.

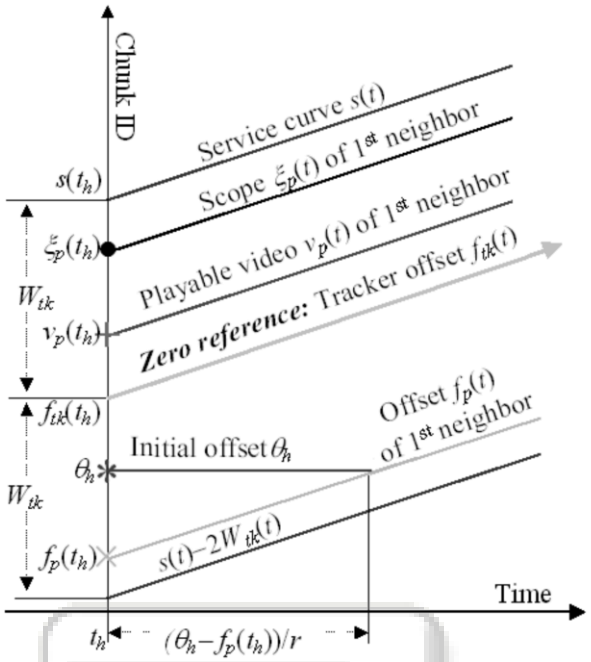

Fig. 5: The startup model

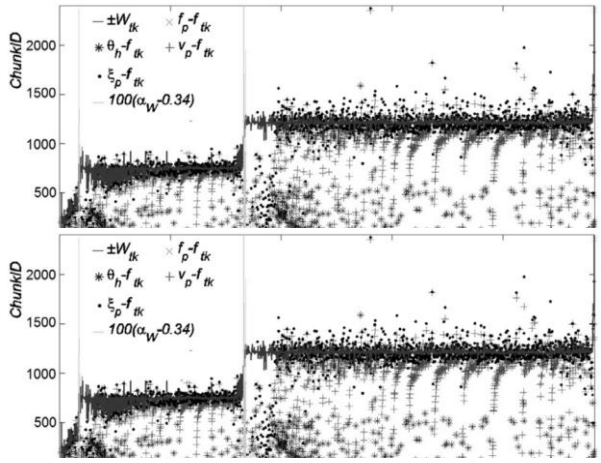

Fig. 6: The measured initial parameters

B. VoD user behavior in P2P streaming media systems

In general, A VoD peer can be classified as *contributor* or *watcher* based on whether the number of ones never increases in bitmap of the peer's BM or not during our observation. In our trace, most peers belong to either contributor or watcher. Less than 6% peers even advertised the abnormal all-zero BMs, the bitmap contained nothing. We guess such disordered behavior ascribed to software bugs, e.g. a user deletes his cache file suddenly. We name such those peers as Zpeer. Fig.12 draws the fractions of different peer groups in our measured channel 1. In fact, the rest two measured channel have the similar results. Those curves confirm that contributors always significantly outnumber watchers, and a stationary process can approximate the fractions. Further, two types of watching modes have been identified. People either watch a movie smoothly until his exit, or see a movie by jumping from one scene to another. We named the former as smooth watching mode and such viewer as smoother, and the latter as the jumping watching mode and that viewer as jumper. Obviously, smoother has continuous 1s in its BM, while jumper has discrete 1s. Table 1 lists the statistics on our trace. We find the majority are smoothers, while the jumpers cannot be ignored. It is different from that "most users always perform some random seeking" (Zheng et al., 2005). As most peers are smoothers, a movie with a larger WI or longer tail in WI distribution in smoothers is usually considered to be more attractive. It means that people watched this movie longer or more people watch the movie. We use probability *pWI(□)* to represent the PDF of WI, which is the fraction of peers whose last "1" in their BMs are at the position □. Fig.13(a) shows Cumulative Distribution Function. Obviously, channel 3 and channel 2 were the most and the least attractive respectively.

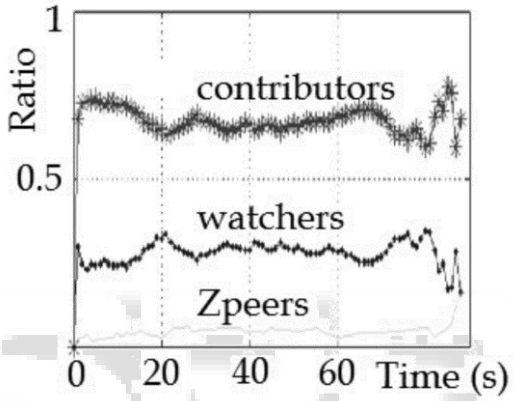

Fig. 7: Role ratios in channel 1.

	Contributors				Watchers			
	Smoothers		Jumpers		Smoothers		Jumpers	
	Peers	%	Peers	%	Peers	%	Peers	%
Channel 1	300	70.9	123	29.1	138	87.3	20	12.7
Channel 2	264	86.8	40	13.1	90	90.1	9	9.1
Channel 3	156	73.2	57	26.8	82	78.8	22	21.2

Table. 1: Number of smoothers and jumpers

1) Measureable parameter watching index in user behavior

For quantitative analysis, we introduce *watching index (WI)* to name the position of the last"1" in a BM, which explains how many chunks a smoother has ever watched. Different from definition in (Yu et al., 2006), we use WI to emphasize the aspects of both time and space. As most peers are smoothers, a movie with a larger WI or longer tail in WI distribution in smoothers is usually considered to be more attractive. It means that people watched this movie longer or more people watch the movie. We use probability $p_{WI}(\theta)$ to represent the PDF of WI, which is the fraction of peers whose last "1" in their BMs are at the position θ. Fig.8(a) shows Cumulative Distribution Function (CDF) $F_{WI}(\theta)=\sum_{k\le\theta} p_{WI}(k)$. Obviously, channel 3 and channel 2 were the most and the least attractive respectively. Besides, online time is defined as how long a peer stays in a channel, and Fig.8(b) shows its CDF. Obviously, distributions of WI over

all channels are significantly different but their online times are very similar. It indicates that WI is strongly related to the video content, while the contributor's online time is nearly independent of what he is currently contributing.

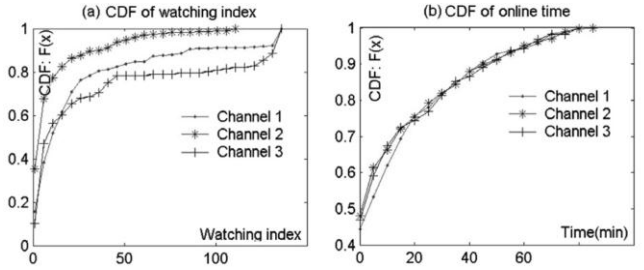

Fig. 8. CDF of WI and online time of contributors

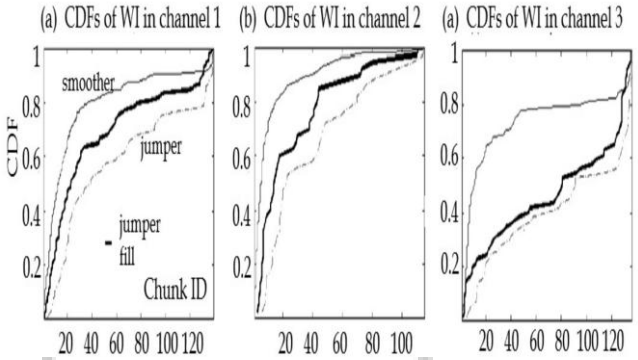

Fig. 9: BM occupancies of contributors

2) User behavior understanding in terms of watching index

WI helps us in better understanding user behavior. Fig.14 shows the CDF of WI for smoothers and jumpers in contributors. The x-axis of each subfigure is the chunk ID or bit positions in the BM. The y-axis is the fraction of the peers. The top curve and bottom curve are of smoother and jumper respectively. The middle curve is the fraction of jumper who has value 1s in its BM at a given bit position. As a peer frequently advertises its BM to others, those subfigures can also be interpreted as the sharing map among VoD peers. Based on this interpretation, we can draw the following conclusions: *i).* although most users are smoother, it may not be good for file-sharing. As lots of people only watch a few chunks, it may lead to overprovision around the initial chunks while under provision for the rest chunks; *ii).*Jumper promotes file-sharing. In each subfigure, the middle curve is significantly below the top curve line. It indicates a jumper contributes more chunks than a smoother. Furthermore, the bottom curve indicates jumpers contribute those chunks with large IDs which smoothers are incapable of sharing; *iii).* Even if jumpers contribute fewer chunks as a whole, their existence is still valuable, as the unbalanced provision resulted from smoothers can be compensated to certain degree by jumpers.

IV. CONCLUSION

In this paper, we presented the study of a P2P streaming media system at different levels of detail. The aim of the study is to illustrate different types of analyses and measurements which can be correlated to reverse-engineer, and further give guidelines to optimizing the behavior of such a system in practice. On signaling message level, we tell about our system crack procedure and reveal the protocol flow and message format. Following that, large-scale measurements are carried out with our network crawlers and mass raw data is captured. On peer behavior level, the startup process of live peer is analyzed in two aspects including initial offset placement and chunk fetch strategies. We discover that the initial offset is the only decision factor to a peer's offset lag (playback delay), and the initial offset selection follows certain proportional placement models based on first neighbor's buffer width or offset lag in PPLive. Once the initial offset is determined, a peer downloads wanted chunks following a TB protocol, which can be depicted by a model of a bunch of piecewise lines. Our measurement proofs that in PPLive most live peers (more than 90%) have seemingly good performance. Moreover, VoD peer's behavior is discussed in user (person behavior. With the help of measurable parameter of WI, we reveal that although majority peers are smoothers, jumpers tend to be the real valuable file-sharing contributor. On

system level, the systematic problems and design concerns on performance, scalability and stability are discussed. Based on the live peer's startup models (PP models and piecewise line model of TB protocol) driven by our trace, we analyze P2P live system's design goals such as the large buffer in peer/small buffer in seeder and self-stability on offset lags, and confirm PPLive tends to really reach those goals. VoD network sharing environment is analyzed in terms of network sharing profile and sharing distribution, and we find the sharing environment is heavily affected by user viewing behavior.

ACKNOWLEDGEMENT

We wish to affirm our earnest acknowledgement to our faculty advisor Prof. Dhaval Jha Department of Computer Engineering, Nirma University for his intuitive and meticulous guidance in completion of this Term paper. We want to express our profound gratitude for his genial and kind co-operation in scrupulously scrutinizing the manuscript and his valuable suggestions throughout the work.

REFERENCES

[1] Ali, S.; Mathur, A. & Zhang, H. (2006). Measurement of commercial peer-to-peer live video streaming, *In Proceedings of ICST Workshop on Recent Advances in Peer-to-Peer Streaming (2006),* Aug. 2006.

[2] Cheng, B.; Liu, X.Z.; Zhang, Z.Y. & Jin, H. (2007). A measurement study of a peer-to-peer video-on-demand system, *Proceedings of the 6th International Workshop on Peer-to-Peer Systems (IPTPS'07),* Washington, USA, Feb. 26-27, 2007

[3] Hei, X.; Liang, C.; Liang, J.; Liu Y. & Ross, K.W. (2007a). A measurement study of a largescale P2P IPTV system, *Journal of IEEE Transactions on Multimedia*, Oct. 2007, Volume 9, Issue 8, (Dec. 2007), pp. 1672-1687, ISSN 1520-9210

[4] Hei, X.J.; Liu, Y. & Ross, K. (2007b). Inferring Network-Wide Quality in P2P Live Streaming Systems, *IEEE Journal on Selected Areas in Communications*, Vol. 25, No. 10, (Dec.2007), pp. 1640-1654, ISSN : 0733-8716

[5] Huang, C.; Li, J. & Ross, K.W. (2007). Can internet video-on-demand be profitable? *Proceedings of ACM*

Sigcomm 2007. pp. 133-144, ISBN 978-1-59593-713-1, Kyoto, Japan, 27-31 Aug., 2007

[6] Li, C.X. & Chen C.J. (2008b). Fetching Strategy in the Startup Stage of P2P Live Streaming Available from http://arxiv.org/ftp/arxiv/papers/0810/0810.2134.pdf

[7] Li, C.X. & Chen C.J. (2008a). Initial Offset Placement in P2P Live Streaming Systems http://arxiv.org/ftp/arxiv/papers/0810/0810.2063.pdf

[8] Li, C.X. & Chen C.J. (2009). Inferring Playback Rate and Rate Resets of P2P Video Streaming Transmissions by Piecewise Line Envelope Approximation. *Journal of China Univ. of Post and Telecom.*, Vol.16, Issue 2, (April 2009), pp. 58-61, ISSN 1005-8885

[9] Li, C.X. & Chen C.J. (2010). Measurement-based study on the relation between users 'watching behaviour and network sharing in P2P VoD systems. *Journal of Computer Networks*, Vol.54, Issue 1, (Jan. 2010), pp. 13-27, ISSN 1389-1286

[10] Lou, D.F.; Mao, Y.Y. & Yeap T.H. (2007). The production of peer-to-peer video-streaming networks, *Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, pp. 346-351,ISBN 978-1-59593-789-6, Kyoto, Japan, 31 Aug. 2007.

