TrustStream: A Secure and Scalable Architecture for Large-Scale Internet Media Streaming

Hao Yin, Chuang Lin, Qian Zhang, Zhijia Chen, and Dapeng Wu

***Abstract*—To effectively address the explosive growth of multimedia applications over the Internet, a large-scale media streaming system has to fully take into account the issues of security, quality of service (QoS), scalability, and heterogeneity. However, current streaming solutions do not address all these challenges simultaneously. To address this limitation, this paper proposes a secure and high-performance streaming system called TrustStream, which combines the best features of scalable coding, content distribution network (CDN) and peer-to-peer (P2P) networks to achieve unprecedented security, scalability, heterogeneity, and certain QoS simultaneously under a unified architecture. In this architecture, raw video is encoded into two layers, namely, the base layer, which contains the most critical media content and is transmitted through a CDN-featured single-source multi-receiver (S-M) P2P network to guarantee a minimal level of quality, and the enhancement layer, which is transmitted in a pure multi-source multi-receiver (M-M) P2P framework to achieve maximum scalability and bandwidth utilization. Heterogeneity is therefore addressed by delivering only the layers that a receiver is able to manage. Security is provided by combining our key distribution mechanism and key-embedding scheme under our proposed S-M P2P topology. We have implemented TrustStream system over the Internet. Deployed by ChinaCache, the largest CDN provider in China, TrustStream has broadcasted several popular live video programs over the Internet. The experimental results demonstrate the advantages and effectiveness of our architecture and system.**

***Index Terms*—Heterogeneity, QoS, scalability, security, streaming Media.**

I. Introduction

WITH the explosive growth of the Internet and society's increasing reliance on multimedia information, we are moving toward a ubiquitous era of streaming multimedia over the Internet: anyone can access the multimedia content on the Internet anywhere, anytime. For this reason, streaming multimedia over the Internet to a large number of users (possibly millions of users) has become an important research topic and application with increasing popularity.

To provide commercial large-scale Internet multimedia streaming, many technical challenges need to be addressed.

First, without copyright management, there is no incentive for commercial content creators to provide multimedia content for Internet streaming; without access control, content providers cannot rake in revenue. Hence, security mechanisms must be in place. However, this is particularly challenging since we need to provide security over an inherently nonsecure system in the Internet.

Second, streaming media has data rate, delay, and packet loss requirements. However, there is no quality of service (QoS) guarantee for huge data transmission of streaming media over the current best-effort Internet. Therefore, QoS assurance poses a significant challenge [29].

Third, an Internet media streaming system should scale well to support a large number of users; in other words, its performance should not be degraded too much as the number of users increases. But achieving scalability is hard since the communication cost and the load of the servers may be extremely high when the number of users is huge, e.g., in millions.

Finally, for multimedia content distribution over the Internet, the heterogeneity of the networks (e.g., different link capacity) and receivers (e.g., different computer processing capability and different QoS requirements) makes it difficult to achieve bandwidth efficiency (due to link sharing in multicast) and service flexibility (needed by different requirements of different receivers) [24], [30].

The success of a large-scale commercial Internet multimedia streaming system will critically depend upon how well it addresses the issues of security, QoS, scalability, and heterogeneity. Existing work has considered a proper subset of these four issues but none could satisfactorily address the four issues simultaneously, which is especially challenging. For example, the current peer-to-peer (P2P) media-streaming technology is seriously limited to providing security and accommodating heterogeneity although it can effectively cope with the issues of scalability and bandwidth bottleneck in the traditional client/server paradigm. In contrast, the technology of content distribution network (CDN) (which is formed by dedicated edge caches for content distribution) is capable of providing security and accommodating QoS, but lacks scalability and suffers from client/server bottleneck and high deployment cost. In addition, the popularity of scalable coding has provided a promising solution for handling heterogeneity, but its current application limits in IP multicast.

Manuscript received December 06, 2006; revised November 01, 2007. First published June 10, 2008; current version published November 26, 2008. This work was supported in part by the National Natural Science Foundation of China under Grant 60673184 and Grant 60873254, in part by the National 863 program of China under Grant 2007AA01Z419, and in part by 973 pre-Program of China (2008CB317101). This paper was recommended by Associate Editor L.-G. Chen.

H. Yin, C. Lin, and Z. Chen are with the Computer Science Department, Tsinghua University, Beijing 100084, China.

Q. Zhang is with the Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.

D. Wu is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: wu@ece.ufl.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2008.927000

1051-8215/$25.00 © 2008 IEEE

To address the limitations of the current technologies and meet with the needs in large-scale media streaming, this paper proposes a novel secure and scalable media streaming system called TrustStream, which builds a new peer-server-peer (PSP) structure to achieve unprecedented security, scalability, heterogeneity, and certain quality of service simultaneously.

The main contributions of this paper are: 1) a novel PSP media streaming architecture, which utilizes the best feature of scalable coding, CDN and P2P, thus possessing the unprecedented capability of addressing all the four critical issues in media streaming, i.e., security, QoS, scalability, and heterogeneity, and 2) a set of security management mechanisms, including a key distribution mechanism and a key-embedding scheme, especially designed for media streaming and combined in our proposed S-M P2P topology. The rest of the paper is organized as follows. Section II discusses related work and highlights the key differences between the existing works and our proposed schemes. Section III describes our TrustStream architecture and each component. In Section IV, we present the implementation details of TrustStream. Section V shows our experimental and system running results to demonstrate the effectiveness of the TrustStream architecture. Section VI concludes this paper.

II. Related Work

Live video streaming is perhaps one of the greatest unfulfilled promises of the Internet. There have been tremendous efforts in the design and experimentation of video streaming systems in the past two decades; there have been no shortage of technical innovations, yet no single system has delivered the expected scalability and service quality [16].

A. Architecture Proposal

To meet the requirements for large-scale multimedia streaming, existing solutions have been mainly focused on coding technology and networking technology. Existing architectures include: 1) multiple description coding (MDC) + P2P [2], [21], mainly for enhancing error resilience; 2) scalable video + IP multicast [12], mainly for addressing heterogeneity; 3) scalable video + P2P [3]; and 4) CDN + P2P [32], mainly for scalability. In addition, recent proposals such as Gridmedia [35], CoolStreaming [36], and PeerStreaming [17] have addressed some issues in security, QoS, scalability, and heterogeneity. But none of the existing architectures consider those four key issues simultaneously. Our TrustStream system presents a unified architecture to addresses these as a whole; the key idea of our architecture is to combine the best features of CDN and P2P as well as using scalable video coding technology with security and QoS enhancement. In short, $\mathrm{TrustStream} = \mathrm{scalable\ video} + \mathrm{CDN} + \mathrm{P2P}$ with security and QoS enhancement.

B. QoS and Scalability

The bottleneck of traditional C/S paradigm is that the total capacity/throughput of the system is limited by the bandwidth of the outgoing link of the server, resulting in low QoS for large-scale streaming. To mitigate this problem, CDN is adopted, i.e., deploying multiple servers or proxies at the edge of Internet to increase total system capacity [29] and provide shortened packet delivery paths. However, CDN does not scale well for a large number of users, especially in the face of a large flash crowd. Commercial CDN's such as Akamai[1] and Limelight Networks[2] are expensive to deploy.

One approach to solve above problem is motivated by the emerging concept of peer-to-peer (P2P) computing and multicast. As an efficient way for content delivery, multicast has been widely researched but its application mainly lies in application layer. Based upon a hierarchical clustering of the application-layer multicast peers, scalable application layer multicast (ALM) [4] supports a number of different data delivery trees with desirable properties. In this paper, we regard application layer multicast as single-source multi-receiver (S-M) P2P. P2P network overcomes the bottleneck around a centralized server with its distributed design and architecture, but also brings a set of technical challenges and issues due to its dynamic and heterogeneous nature. The implementation of ESM [10] and CoolStreaming [36] marked a new era for P2P real-time streaming systems. However, these P2P solutions achieve scalability at the cost of losing manageability; thus they could not guarantee QoS and address security issues well. Meanwhile, current solutions only use S-M P2P, i.e ALM or M-M P2P, i.e pure P2P in CoolStream, separately. To address this, we combine the best features of CDN and P2P in our PSP networking. We employ CDN-featured S-M P2P to guarantee QoS and facilitate security management, while deploying a pure multisource multi-receiver (M-M) P2P framework for media delivery among peers to achieve maximum scalability.

Recent proposals [25], [32] try to directly combine CDN and P2P networks to disseminate media content faster and respond more quickly to requests. However, it is not clear how to merge those networks and support security in the hybrid CDN + P2P networks. To address this, we adopt scalable coding to merge the best feature of P2P and CDN, and deliver two layers of media content separately through S-M P2P and M-M P2P. Our S-M P2P is actually organized in an ALM mesh tree; but specifically, we deploy some fixed server nodes in a logical S-M P2P network and all nodes are organized in a hierarchical multicast tree. Furthermore, in M-M P2P management, we adopt a bandwidth-based metric in gossip protocol to alleviate congestion at certain popular nodes. The details can be found in Section III-B and III-D.

C. Security

Security issues in media streaming systems include: 1) content confidentiality; 2) content integrity; 3) content availability; 4) user authentication; and 5) digital right management (DRM). Since problems 4) and 5) have been widely explored, this paper focuses on problems 1–3.

Most key distribution schemes available are based on a media-independent approach, i.e., the generation of a new key is triggered by time or an event independent of the media content; these schemes cannot meet the requirement of P2P streaming because: 1) users in the P2P network may view

[1] [Online]. Available: http://www.akamai.com

[2] [Online]. Available: http://www.limelightnetworks.com

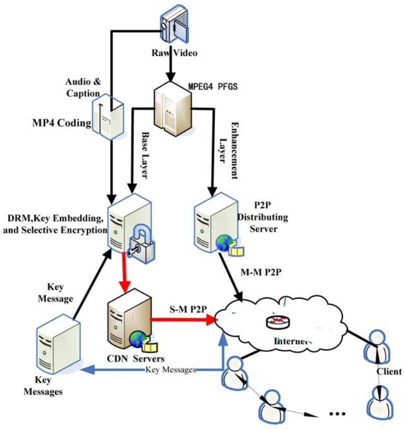


Fig. 1. Framework of TrustStream.

different content/frames at the same time and 2) the overhead of key updating is too high. To mitigate these limitations, we take a media-dependent approach, i.e., keys are bundled with media content packets. Specifically, we use two types of keys, namely, session keys and cluster keys; generation of the session keys is time-driven (triggered by synchronization markers in media content) while generation of the cluster keys is event-driven (triggered by events such as joining or leaving of a node). Compared to the media-independent approach, our approach significantly decreases communication overhead for key updating, and improves security by imposing rules for embedding keys in media packets.

Centralized key management is applicable for small-scale Single-source Multi-receiver multimedia multicast applications. Different network structure will incur different communication overhead in key distribution [23]. But even with the hierarchical tree, the overhead for C-FT and LKH [28] is still large in a large scale P2P network (e.g., with millions of users).

To mitigate this problem, we take a decentralized key management approach; our idea is to use a cluster-based hierarchical tree. We use session keys to encrypt media content and use cluster keys for the distribution of session keys. In our scheme, when a node leaves or joins, we only need to update the cluster key in the cluster of the leaving/joining node; the communication overhead is $O(l)$, where l is the number of members in a cluster, usually a constant. In contrast, the existing centralized schemes need to perform key update for all related users, which is $O(n)$ or at least $O(\log n)$.

Furthermore, we combine our key management scheme [33] and data embedding scheme called SMDE [34], which is error resilient and transparent for rate adaptation. Our security schemes are deployed over a scalable hierarchical S-M P2P structure, and combine a novel key distribution mechanism and a key-embedding scheme to achieve confidentiality, integrity, and availability through a media-dependent approach. Our scheme has the advantage of reduced rekeying overhead (compared to centralized key management) and better central management (compared to existing decentralized key management schemes) by using fixed server nodes as trusted rekeying cluster leaders. The details will be presented in Section III-C.

D. Heterogeneity

There are two kinds of heterogeneity, namely, *network heterogeneity* and *receiver heterogeneity*. Network heterogeneity could make different users experience different packet loss/delay characteristics. Receiver heterogeneity means that receivers have different or even varying latency requirements, visual quality requirement, and/or processing capability. Multicast is usually used for media streaming due to its efficiency and scalability. But the sharing nature of multicast and the heterogeneity of networks and receivers sometimes present a conflicting dilemma.

In order the address the above two problems, many researches proposed the idea by combing the multicast and scalable coding. Scalable video [18], [19], [29] is used so that different users with different link bandwidth can subscribe to different sets of multicast video streams. However, existing scalable video streaming systems such as CoopNet [22] use a Client/Server paradigm thus has not yet unlocked the potential of the P2P technology. Multicast scalable video has not fully utilize the scalability of a pure P2P network.

Our approach is different in that it adopts layered coding in S-M P2P and M-M P2P, resulting in robustness and high flexibility of network topology. Compared to the multiple description coding (MDC) approach [15], [31], our approach generates two video layers of different priority/importance, and provides better QoS for a layer with higher priority and importance, resulting in higher efficiency in utilizing the peer resources.

III. A Secure and Scalable Architecture for TrustStream

In this section, we present the PSP architecture for TrustStream, a novel secure and scalable system for large-scale media streaming application over the Internet. We first introduce the architecture, and then we present our mechanisms for scalable hierarchy topology of S-M P2P, security management and M-M P2P membership management.

A. Architecture

Fig. 1 shows the overall framework of TrustStream. Our simplified progressive fine granularity scalable (PFGS) generates two layers of streams: one of very low bit rate, called "base layer," with essential but most important information of media content; while the other of much higher bit rate, called "enhancement layer", with only enhancement to playback quality. A receiver can decide whether it wants to receive certain enhancement layer pieces thus progressively improving video quality by obtaining more trunked enhancement layer pieces; heterogeneity is therefore handled by delivering only the layers that a receiver can manage. The base layer content is encrypted by embedding copyright information with our proposed video data embedding codec [34], and then by applying a selective encryption to it, which can prevent illegal users from accessing the content. After this, the encrypted base layer content is sent out in the S-M P2P framework, along with the key messages, while the enhancement layer is sent in the M-M P2P framework. The above operations are carried out on the server.

The client in TrustStream will then receive two layers of streams, that is, the base layer stream and the enhancement layer stream. The base layer is necessary for media decoding, providing essential and acceptable playback quality. At the same time, the key message taken with it can help to update the session key periodically. The security management message and media content are "re-assembled", namely, the enhancement layer can only be decoded if the base layer is available. In other words, it is useless for the client if the base layer is lost. With the separation from data transmission, security management is facilitated in our structure. The enhancement layer is optional, which mainly improves the quality relying on a stream of much higher bit rate. Client can get appropriate amount of enhancement layer content according to its network bandwidth, resulting in corresponding improvement of playback quality.

From an overview, the streaming content transmission of TrustStream follows the following process: First, the PFGS encoder P_{sc} maps a given frame F_k into layered codes L_{kb} for base layer, L_{ke}^i for trunked enhancement layer pieces, where $i = (1, 2, \ldots, n)$

$$P_{sc} : F_k \mapsto \left\{ L_{kb} + \sum_{i=1}^{n} L_{ke}^i \right\}. \tag{1}$$

Then the encryption process E_p conducts a copyright and selective encryption over base layer content L_{kb} and produces the encrypted base layer EL_{kb}

$$E_p : L_{kb} \mapsto EL_{kb}. \tag{2}$$

Finally, along with the key information Key_k, the decoder D_{sc} at the receiving end maps EL_{kb} and all received enhance layers $\sum_{i=1}^{m} L_{ke}^i (m \leq n)$ to reconstruct the initial frame as $\wedge F_k$

$$D_{sc} : \left\{ EL_{kb} + Key_k + \sum_{i=1}^{m} L_{ke}^i \right\} \mapsto \wedge F_k (m \leq n). \tag{3}$$

In our architecture, the best feature of P2P and CDN networks are combined together by delivering the media content from source server in CDN-featured networks to edge servers and then transmitting the content between each P2P peers. In this way, CDN guarantees the QoS and P2P enhances the scalability of streaming system. The S-M P2P base layer transmission has a scalable hierarchical multicast structure upon which security is managed. The pure M-M P2P enhancement layer transmission adopts a gossip-based P2P membership management, and achieves scalability, reliability and load balancing with acceptable overheads. In Sections III-B–D we will illustrate in detail how we achieve this design.

B. S-M P2P Hierarchy for Base Layer

We use CDN nodes as servers in a logically S-M P2P framework. The idea of multicast is adopted in the S-M P2P implementation by arranging the clients in a multicast mesh topology while maintaining several CDN-featured servers to guarantee the source streaming. From traditional wisdom, a multicast topology from source to receivers needs to be a tree, i.e in [27] there is a quite simple, robust and effective tree-based P2P multicast protocol called Chunkyspread, and we move further in our implementation by proposing a novel distributed algorithm to construct a netlike and treelike multicast graph as the S-M P2P topology, upon which the data transmission and security schemes are applied. Our protocol arranges the set of members into a hierarchy. It implicitly defines the multicast overlay data paths. When members join or leave, the hierarchy is maintained accordingly.

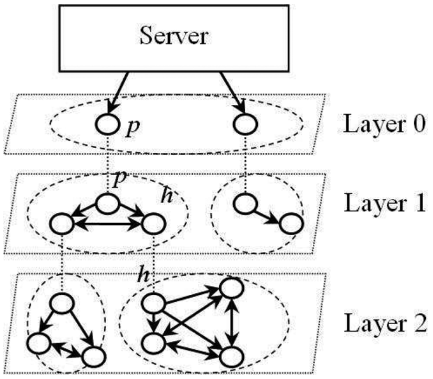


Fig. 2. Data Forwarding in our secure S-M P2P protocol.

TABLE I
PROCEDURE: NEW-LEADER-ELECTION $(k, CL_j(L, S))$

```
1. NEW Node $k$ sends a probing message to all members in $CK_j(L, S)$;
2. FOR all $i \in CL_j(L, S)$; Compute value $w_i(k) = (x_i h, y_i h)$;
3. Set distance $r_i(h) = m_1 x_i h + m_2 y_i h$; Compute $\overline{r}(k) = \frac{\sum_{i=1}^{n} r_i(k)}{n}$;
4. IF node $k$=the first/second node in empty cluster $CL_j()$;
5. Set $k$ as the leader or sub-leader directly;
6. ELSE if $\overline{r}(k) > \overline{r}(L)$ Elect $k$ as the leader;
7. ——ELSE if $\overline{r}(k) > \overline{r}(S)$ Elect $k$ as the sub-leader;
8. Update $CL_j(L, S)$ and send feedback to server.
```

The hierarchy consists of members assigned to different layers as shown in Fig. 2. The top layer of the hierarchy is layer L_0 which contains carefully chosen CDN-featured fixed nodes. Besides the fixed nodes in the top layer which are appointed by the server, nodes in other layers are organized in an ad hoc manner to form an optimized topology in a top-down manner from the layer L_0. Members in each layer are organized into clusters. Each cluster has a leader, called "cluster leader." The cluster leader is also a member of the corresponding cluster in the higher layer. The cluster leader should have the maximum local performance (bandwidth, net utility, CPU ability, etc) and the minimum average distance (RTT) to other members in the cluster. Each cluster also has a subsidiary leader who is getting ready to take the responsibility of current leader when it leaves. Specially, since the data source is Server, it can be considered as the leader of layer L_0. Table I shows how a cluster is formed and how the leader and sub-leader are elected.

The metric used here is $w(h) = (x_h, y_h)$, where x_h denotes the estimated end-to-end bandwidth and y_h denotes the end-to-end latency. Set the metric $r(h) = m_1 x_h + m_2 y_k$, where m_1 is some proper chosen negative and m_2 a positive number. $r_i(h)$ denotes the distance from node h and node i. $\overline{r}(h)$ denotes the average distance from node h to all nodes in its cluster, $\overline{r}(h) = (\sum_{i=1}^{n} r_i(h)/n)$. The leader is the node that has minimum average distance away from other nodes in the cluster. Set

R as the boundary of a cluster where all $r(h)$ between nodes are within the value of R, otherwise it is outside the cluster.

1) Topologies Management: The member hierarchy defines both of the control topology and data overlay topology. In the control topology, as illustrated in Fig. 2, each member h exchanges maintenance-messages (mainly used to maintain the hierarchy) periodically with other members in the same cluster. The data topology is defined by the following rule: The source member is *Server*, which sends data packet directly to all the members of L_0. Consider an arbitrary member h, who receives the data packets from member p. Then p and h must belong to the same cluster C_k in a certain layer L_i $(k > i)$. Member h will forward the data packets to all other members of cluster C_k in a P2P routing scheme [26], if and only if h is the cluster leader of C_k. But, in Layer C_0 the *Server* directly forward data packets to all the members instead.

2) Join: When a new member tries to join, it firstly contacts the *Server*. The *Server* should conduct the CA verification and then sign a time-limited label for its identity. Meanwhile the *Server* should also send a list of recommended leaders to the new member. Then the new member sends joining requests to these leaders and wait for the replies till it achieves an acceptance of the most suitable cluster leader. After that the new member completes its joining process.

3) Maintain: If the performances of other members in the cluster are better than cluster leader, the leader should be replaced by a better member to improve the overall performance of the whole system. In order to guarantee the QoS of media playback and reliability of SK distribution, the cluster sub-leader also maintains the cluster information by periodically communicating with the leader so that it can recover the relationships with the nodes in higher layers and keep the organization of the cluster as soon as leader fails.

4) Leave: When a member leaves the group, it sends a *LeaveMessage* to the *Server* and its neighbor members.If the leaving member is a leader, the cluster subsidiary leader will become the new leader and join the higher layer cluster where the leaving leader ever lies. If the leaving member is just a member, the leader should report to the *Server* to expire its label. Unexpected leaving would be detected by the cluster leaders/sub leaders in means of periodically sending a query and waiting for a reply control message.

C. Security Management for Base Layer

Confidentiality in multicast system is usually achieved by encrypting the content using an encryption key, known as the session key (SK) that is only known by the content provider and all legitimate group members [8]. However, it is not an easy task to deliver the SK to all the members securely because the group membership is most likely dynamic with clients joining and leaving the group from time to time. Notice that, once a member is not in the multicast session, e.g., before he joins or after he leaves the session, he should not be able to access the media content. In other words, the SK needs to be updated once a member joins or leaves the session. In our scenario, the key management and distribution scheme have the following three security properties.

TABLE II
NOMENCLATURE IN KEY DISTRIBUTION

SK	Session Key	shared by all group members and KMS; used to encrypt/decrypt streaming media;generated by KMS periodically
CK	Cluster Key	Shared by each cluster; used to encrypt/decrypt SK; generated by the cluster leader
KMS	Key Management Server	Generates SK; Distributes SK; Manages C_0
SC	Secure Channel	Established by using Public key and Private key between members and leader

1) Forward Secrecy: to ensure that an expired member cannot access the new SK after he leaves the group.
2) Backward Secrecy: to ensure that when a new member joins the group, he cannot access previous media contents. Without this property, a client can first receive and store the multicast data, and then he joins the multicast session and gets a SK, and tries to use the SK to decrypt previous media content.
3) Collusion: to prevent expired members from working together and sharing their individual piece of expired SK information to regain access to the new SK [8].

Table II illustrates the function of SK, CK, KMS, and SC.

By using key management and distribution algorithm, we implement a novel secure CDN that solves the following issues:

1) Confidentiality: In each cluster there is an election and intendance mechanism, by which cluster members can elect two trusted nodes as cluster leaders (one acts as the security manager and media content source in this cluster and the other as the assistant or backup) and dismiss the ones who lose confidence. The content is encrypted by using an encryption key. Only authorized users can get the encryption key from the leader to correctly decode the incoming base layer content. In this case, server does not need to carry out a global key update and distribution process, which can effectively eliminate the potential bottlenecks on the server.

Key management is added to the CDN, there are two aspects: *Generation and distribution of SK*: KMS generates a new SK periodically. At beginning, KMS generates a new SK, which is used to encrypt/decrypt the streaming media content in the next period, and then sends the updating message of SK to all the members in C_0, encrypted with the CK_0(Cluster Key of C_0. As illustrated in Fig. 2, consider an intermediate member, h, that receives the refresh message from another member p. Then h decrypts the SK with the CK shared by the cluster when p and h belong to the same layer L_i. If h is also in another layer, $L_k(k \succ i)$, h must be a cluster leader of one cluster, C_k. Then h re-encrypts the SK with the CK shared by C_k and forward the SK refresh message to the members in C_k.

2) Generation and Distribution of CK: The cluster leader updates CK when cluster members change and then distributes it to the cluster members through secure channels. When a new member joins the cluster, it should establish a secure channel with the leader. When the cluster leader leaves, the subsidiary leader should establish secure channels respectively with cluster members, and then generates a new CK and distributes it.

3) Integrity: Every member has a pair of keys, i.e., a public key and a private key; the public key is also maintained in the KMS. When a member sends some message, it adds its digital signature to the message; the digital signature is signed with the private key of the sender. When the receiver receives the message, it can authenticate the message by the public key of the intended sender, which can be obtained from the KMS.

4) Availability: Since updating message of SK is distributed along with the media streaming over the error-prone network, it is important to guarantee the reliability of the SK distribution. In TrustStream, we set a subsidiary leader in each cluster to solve this problem. If the leader leaves or collapses, the subsidiary leader can easily join the higher layer and get the media content by using the backup information from the leader periodically. We also distribute the same updating message of SK several times at one update interval. If a member losses the updating message, it can get it by the redundant updating message along with the coming media content.

We further combine the above key management scheme with our novel data embedding scheme named as SMDE [34], which is of error resilience and transparency for adaptation mechanism. In this scheme, the key messages are embedded in the host video signal and distributed to the authorized users. Using embedded data to convey key information is able to achieve added security and reduce bandwidth resource consumption. The combination of key distribution mechanism and key embedding scheme could provide secure access control for adaptive video multicast applications.

D. M-M P2P Management for Enhancement Layer

In our system we adopt a gossip-based [20] protocol which provides good scalability and reliability properties to manage the P2P members for the transmission of enhancement layer content.

To decrease the overhead of gossip schemes, we could consider two issues, one is to minimize the neighbor list (which we call membership knowledge) one node would have to guarantee the transmission and the other is to decrease the probability for nodes to receive redundant messages. The latter can be done by assigning different weights to neighbors. A message floods to neighbors that have small weights and gossips to neighbors that have large weights.

We introduce a parameter M here to denote the membership knowledge one node holds to randomly send packets to gossip targets. Table III illustrates how this M can be formed and content can be obtained.

TABLE III
GOSSIP MEMBERSHIP MANAGEMENT AND CONTENT DISTRIBUTION

1. A new node l randomly sends a request P for content C;
2. IF m =the node who receives the request P;
3. FOR all $n \in M_m$; if n holds C,send (C, l);
4. m creates c additional copies of P //to backup for failures;
5. FOR $(i = 0; i < c; i + +)$; choose randomly $n \in M_m$;forward (P, n);
6. FOR n =the node who receives a forwarded request P;
7. IF n holds C; send (C, l);
8. IF with probability $p = \frac{1}{1+M_n}$; store (L, M_m), (n, M_L);
9. ELSE, n randomly forwards request P to nodes in M_n until one receives it.

Now we could build a random directed graph topology of the system: there is a directed arc from x to y whenever y is in the M of x and when new node joins, it creates a random number of additional arcs according to the above mechanism. This forms the basis for broadcasting messages across the group, by enabling each member to propagate messages to all or to a subset of those members with its M in the connected transmission graph. Let W_n denote the total arcs in directed graph which model the whole system with n members. Based on this strategy, we could get,

$$W_n = W_{n-1} + \frac{W_{n-1}}{n-1} + c + 1 \tag{4}$$

$$W_n \approx cnlgn + nlgn. \tag{5}$$

Then the average out-degree of each node is

$$E_n = \frac{W_n}{n} = (c + 1)lgn. \tag{6}$$

According to the theorem in [14] that: if there are n nodes, and each node gossips to $\log n + k$ other nodes on average, then the probability that everyone gets the message converges to $e^{-e^{-k}}$ which is quite near to 1.

Similar idea is presented in the design of SCAMP [11] which operates in a fully decentralized manner and provides each member with a partial view of the group membership. We have optimized the scheme by combining the content searching in the member subscription procedure and adding the metric of bandwidth to avoid convergence of content searching in limited hot nodes by making each node to measure the bandwidth of the gossip target and dynamically get content from the available target with comparatively higher bandwidth. Load balancing can be achieved by moving nodes to obtain content from other nodes when the bandwidth of hot nodes becomes smaller than some other nodes.

Lastly, although the layered coding facilitates our security and QoS management, the separation of base layer and enhancement layer may bring about the issue of synchronization of the layers. This problem can be solved by pre-fetching the base layer in the buffer and waiting for the proper enhance layer frame to come to re-construct as a whole. By pre-fetching, the system can avoid the loss of a substream upon a node disconnect even when the replacement time is nonnegligible. Once received the base layer, the clients could obtain a fundamental quality of the streaming media and improve its quality if more according enhancement layer pieces are received later.

Media synchronization refers to maintaining the temporal relationships within one data stream and amongst various media streams and the essential part of any media synchronization mechanism is the specifications of the temporal relations within a medium and between the media. The methods that are used to specify the temporal relations include interval-based, axes-based, control flow-based, and event-based specifications. We have discussed this issue in our former work [29] and more details could be found in [6].

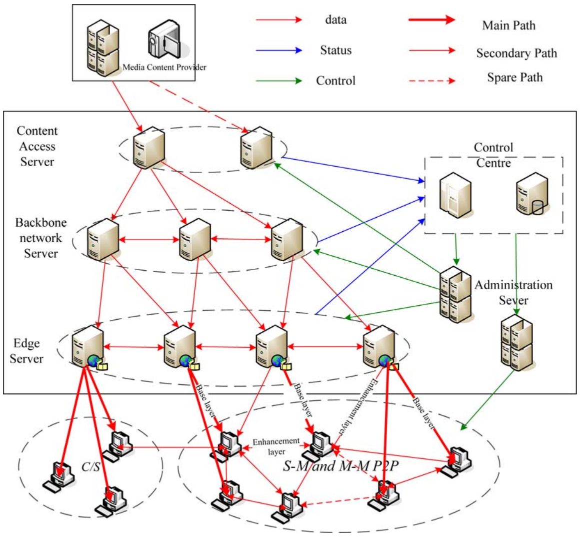


Fig. 3. Implementation framework of TrustStream.

IV. Implementation

Our implementation framework for TrustStream is shown in Fig. 3. Between media content provider and end users, we deploy three levels of CDN-featured fixed server nodes and some management servers. The edge servers are deployed nearby peer clusters, from which the base layer content is transmitted in a S-M P2P network while the enhancement layer content is transmitted in a M-M P2P network. Media content is first distributed among the carefully deployed trusted servers and finally reaches edge servers, from which end users can choose to obtain the media content either in C/S mode or P2P mode. Those carefully-deployed servers can guide streaming traffic to achieve overall traffic optimization and Global Sever Load Balancing (GSLB), and conduct access control and key distribution. Strategically deployed fixed server nodes around the Internet enable end users to obtain streaming video from one of the nearby servers to reduce the end-to-end delay and overall network congestion. QoS is guaranteed by delivering the base layer content directly from the CDN-featured nodes in S-M P2P to ensure basic video quality even in time of peer failure. Meanwhile, the utilization of M-M P2P networks around those servers facilitate peers to freely transmit in a pure P2P protocol to achieve maximum scalability.

In detail, from the server side, TrustStream is divided into four parts: content source, content management, network management and mid nodes. The raw video data that comes to the encoder are divided into the base layer stream and the enhancement layer stream. A central management server deals with the authorization of clients, key managements and conducts encryption on base layer stream. Then the baser layer stream is sent to the authorized users in our secure S-M P2P network and the enhancement layer stream is sent in a pure M-M P2P way in network of either wireless or LAN's.

Fig. 4 is the client software architecture of TrustStream, which consists of network operation, P2P protocol management and the playback. The system uses the message-driven mechanism, including proactive messages (the timer evens) and passive messages (requests from other peers). Two kinds of buffer pools are used to store data in base layer and enhancement layer. Besides the buffer control, the P2P protocol part also contains: P2P membership management, event and message dispatcher, key management, data source scheduler and the timer. In the playback process, the player would select the data according to the synchronization of the audio and video data, from base layer or enhancement layer in the buffer. Layered decoding would be conducted on video data while common decoding would be used for audio data. During the decoding process, decrypting is conducted with the given SK. In this way, the client side can play back a real-time and decrypted media stream.

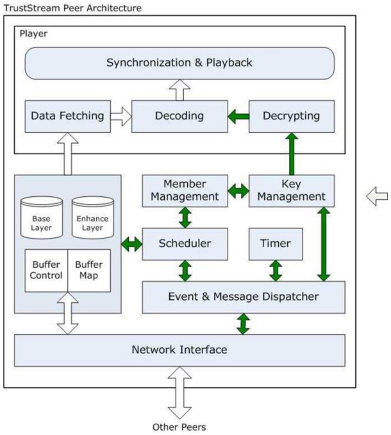


Fig. 4. TrustStream client framework.

To promote process efficiency and reduce system overheads, we adopt PFGS encoding and decoding for base layer and enhancement layer. PFGS uses as many predictions from the same layer as possible to increase coding efficiency; and PFGS keeps a prediction path which always uses prediction from a lower layer in the reference frames (for error recovery and channel adaptation)[18], [19]. To get the tradeoff between coding efficiency and error-resilience, we have simplified the coding operation without decreasing the coding efficiency after our deduction of enhancement layer coding process in Inter mode.

Table IV shows some system settings of TrustStream, including the testbed scale, CPU utilization, supporting bit rate, etc. We will further specify TrustStream running parameters in the performance evaluation part.

V. Performance Evaluation

A. Source Coding

We have implemented TrustStream system and benchmarked its performance in both simulation environment and real network.

For the encoding of TrustStream, we adopt a frame-based PFGS coding to generate the base layer bitstream with a bit rate of 128 kbps and one enhancement layer of around 300 to 400 kbps.

Fig. 5 gives the Peak Signal to Noise Ratio (PSNR) of the decoded video. For all the test sequences, we observe a consistent quality improvement with the increase in the video bitrate of PFGS. As shown in Fig. 5, though PFGS is not as good as

TABLE IV
PARAMETERS OF TRUSTSTREAM SYSTEM

Member maintaining interval	$2 \sim 4$ seconds
Buffering time	$5 \sim 15$ second
CPU utilization	$\prec$ 30%
Supported user magnitude	100,000 users
Supported bitrate	$128kbps \sim 2Mbps$

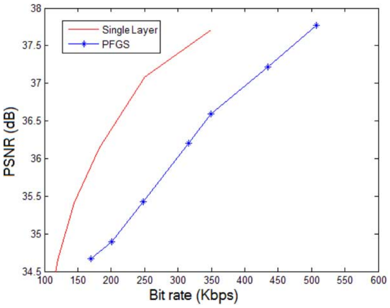
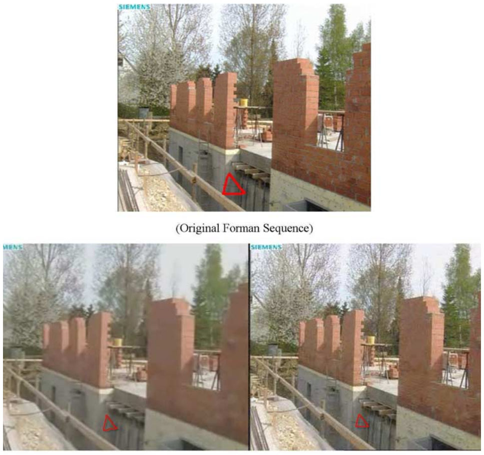


Fig. 5. Bit rates of single-layer and PFGS on "highway-CIF."


(Original Forman Sequence)

(WM encoder 9.0)

(TrustStream Encoder)

Fig. 6. Encoding performance comparison (original; WM; Truststream).

unscalable video coding on coding efficiency, it still provides acceptable video quality and bandwidth requirement. Considering the superior layered feature needed by our structure which cannot be provided by existed traditional single layer coding, PFGS is thus an ideal choice for efficiency and function. Actually, as is shown by the Foreman sequences encoding performance in Fig. 6, compared with WM encoder, the PFGS-based Truststream encoder provides satisfactory performance.

B. QoS

TrustStream has been implemented in real network and broadcasted several nationwide popular live video programs all over China, including the national Spring Festival Show 2007, the celebration for the traditional Chinese New Year day on February 17, 2007, when hundreds of thousands of users viewed the live video program from Internet. TrustStream is adopted by ChinaCache, the largest CDN provider in China, which deployed more than 150 servers and a total of 18 Gbps bandwidth around China last year. As Fig. 7 shows, 24 core servers were deployed to establish the system in the two core ISP's in China, among which, 6 are in China Netcom (CNC), and 18 in China Telecom (CHN). This figure also shows how the live streams are piped among those servers. The rest cache servers were deployed nearby users. At peak time, we attracted around 42,850 views from users spread from over 20 provinces in China, and four countries overseas(mainly from the districts where oversea Chinese people live). Among them, 30,088 simultaneous views obtained content from our deployed CDN servers and the total server bandwidth achieved 6.7 Gbps. According to our traces analysis, at peak time, the P2P traffic accounts for around 30 $\%$ of the total bandwidth.

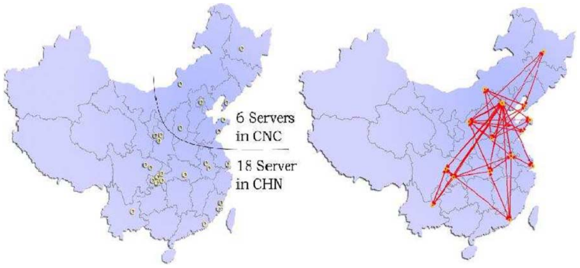


Fig. 7. TrustStream server deployment around China.

TABLE V
COMPARISON OF SPRING FESTIVAL LIVE 2006 AND 2007

Year	Source Bit Rate	Peak Bandwidth	Average Online Time
2006	128 Kbps	5.35 Gbps	559.4 sec
2007	225 Kbps	6.70 Gbps	1451 sec

Table V demonstrates the statistics comparison of Spring Festival Live in 2006 (traditional media streaming) and 2007 (TrustStream). As the data shows, the average source bit rate of our system can reach 225 kbps, thus being capable of providing satisfactory live video quality.

Similar to the paper [9] that quantifies QoS of skype upon Call duration, we define the average viewing time, T_v to indicate how much the user is satisfied with our streaming service. Besides the minimal content difference (i.e all are popular contents), the more users feel satisfied with our service, the more time they tend to view the media online. The 6.70 Gbps peak bandwidth and 1451 sec average user online time demonstrate our system is performing with good user satisfaction. With the improvements of quality, users of our system tend to stay in viewing much longer duration time, i.e 160% increase, than they used to be in former year 2006.

For another metric of start-up delay in media service, as paper [13] reports, for the popular IPTV system PPlive, the player pop-up delay is generally 10 to 15 seconds and the player buffering delay is around 10 to 15 seconds. Therefore, the total start-up delay is around 20 to 30 seconds. And some less popular channels may have a total start-up delays of up to 2 minutes. In comparison, with our carefully deployed servers and system structure, during our whole program, as our log analysis shows, the average user start-up delay is within 15 seconds.

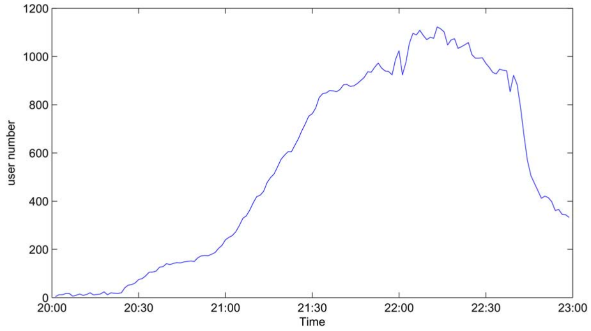


Fig. 8. Online user number on one server.

C. Scalability

Fig. 8 shows some real network data in one server node. At 22:10 pm the number of users viewing the program through the observed server arrives at the maximum. Accordingly, the total traffic transmitted from the same server node, achieves maximum, indicating a single server can support a scale of 1100 peers and the system can well guarantee the stability of source signal from the server node.

To theoretically quantify the scalability of our system, we define $Bd_{\text{incr}}()$ as bandwidth magnifying multiple, i.e how many times bandwidth is increased from input to output. This parameter can describe to what scale our system can support users, with the increased bandwidth in contrast to the former single server in C/S structure. The total system bandwidth magnifying multiple would be: $Bd_{\text{incr}}(\text{system}) = Bd_{\text{incr}}(\text{live broadcast}) * Bd_{\text{incr}}(P2P)$. Here $Bd_{\text{incr}}(\text{live broadcast})$ means the bandwidth magnifying multiple provided by server parts of our framework in Fig. 3. Assuming there are n super nodes in the backbone network level, each of which are responsible for handling m edge nodes. One edge node, which may compose of several edge servers and usually have a total capacity of 1 Gbps, can serve k users. According to our organization of multicast tree, we have

$$Bd_{\text{incr}}(\text{live broadcast}) = \frac{\text{Throughput}_{\text{out}}}{\text{Throughput}_{\text{in}}} \tag{7}$$

$$= \frac{T*n*m*k}{T*(n+\frac{n*(n-1)}{2}+n*m)} = \frac{k}{1+\frac{n+1}{2m}}. \tag{8}$$

As $0 < (n+1/2m) < 1$, the $Bd_{\text{incr}}(\text{live broadcast})$ is around $((k/2), k)$.

In our direct broadcasting for Bill Gates' live speech at Peking University in year 2007, a total 3.9 Gbps peak bandwidth were utilized. As Table VI shows, our actual bandwidth magnifying multiple arrives at 484:1, which well demonstrates the above theoretical analysis. In comparison, another dominating live video provider in China only provided 1.3 Gbps peak bandwidth and a 250: 1 of bandwidth magnifying multiple when broadcasting the same program.

Then with the participation of P2P network, we set k as the ratio between the average client uploading bandwidth and video program rate. Suppose there are a_i peers at level i, where the depth of i ranges from 0 to n. Then the total number of clients would be

$$S_n = \sum_{i=0}^{n} a_i = a_0 * \frac{k^n - 1}{k-1} \tag{9}$$

thus

$$Bd_{\text{incr(P2P)}} = \frac{k^n - 1}{k-1}. \tag{10}$$

TABLE VI
COMPARISON OF LIVE VIDEO BROADCASTING PERFORMANCE

System	Bit Rate	Peak Bandwidth	Magnifying multiple
TrustStream	106 Kbps	3.9 Gbps	484:1
Other Provider	137 Kbps	1.3 Gbps	250:1

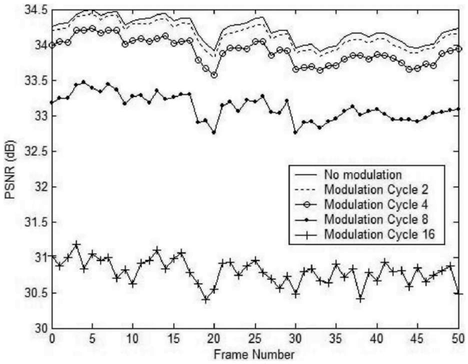


Fig. 9. Effects of security modulation over video quality.

As an example, in one scenario of broadcasting one program with our system, the video program rate is 200 kbps, the assigned server bandwidth is 100 Mbps (capable for 500 connections), the average clients uploading bandwidth is 400 kbps and the maximum cluster levels are 5. Then the average number of users that could be supported would be 15000. The P2P network has helped to increase the bandwidth for 50 times and for the whole system, it has totally scaled to 7500–15000 times.

D. Tradeoff and Security Overhead

We examined QoS and Security trade-offs by analyzing the negative impact of security management on video quality. Fig. 9 shows the effects of 200 bits data embedded in sequence ”Dinosaur” with different modulation cycle and the PSNR of frames for varying modulation cycle C at the receiver. When the modulation cycle is 4, the PSNR (represented by the bold curve with circles) falls by only 0.2 to 0.3 dB, which is tolerated by contrast with the fall of PSNR when modulation cycle equals 8 or 16. From this figure we can find that the modulation cycle with 4 is a tradeoff between good quality of video and high detection accuracy.

The overhead for security management is analyzed as follows.

- *Secure Channels*. The number of secure channels inside the tree is $L-1$, where L denotes the size of a cluster. KMS should generate a new CK_0 when members change, and send it through the secure channels to each of the members in cluster C_0.

TABLE VII
OVERHEAD OF ENCODING AND DECODING

	Without Enc/Dec	With Enc/Dec	Overhead
Encoding	50.4 fps	49.1 fps	2.64 %
Decoding	112.8 fps	112.2 fps	0.54 %

- *Overhead of CK updating*. When cluster members change, CK refresh messages are distributed through the secure channels. Since the size of each cluster is small enough and independent of the group size, the overhead can be considered as $O(1)$.
- *Overhead of SK re-encryption*. The SK needs to be decrypted and re-encrypted by leaders between two clusters along the path, therefore the number of re-encryption operations is $O(log_L N)$. Here L is the average cluster size and N is the total number of nodes. But to a single leader, this overhead can be ignored.
- *Overhead of cluster maintaining*. Members exchange maintenance-messages with others in the same cluster. Since the size of cluster is small enough and independent of the group size, the overhead is $O(1)$.

As for the management overheads in real maintenance, from Table VII we can see that in our simulation experiment with "forman" streams, the encryption overhead is negligible, i.e., only a percentage of 2.64 for encoding and 0.54 for decoding overhead.

VI. Conclusion

There is ever-increasing demand for multimedia content over the Internet, but current streaming solutions do not suit well with large-scale applications. Challenges mainly lie in four aspects: 1) security; 2) scalability; 3) heterogeneity; and 4) QoS.

In this paper, we propose the TrustStream, a novel, secure and scalable media streaming system to address the above challenges in a unified architecture. Our TrustStream combines the best features of CDN and P2P networks, which are merged by scalable coding, to achieve unprecedented security, scalability, accommodation of heterogeneity, and certain QoS simultaneously. First, security is provided by combining the key distribution mechanism and key-embedding scheme under our proposed Secure S-M P2P. Second, scalability is achieved by utilizing the pure M-M P2P in our enhancement layer content distribution. Third, heterogeneity is addressed by delivering only the layers of content that a receiver is able to manage under its resource constraints. Fourth, quality of service is achieved by the CDN-featured S-M P2P.

Our system was implemented in early 2007 and has broadcasted several popular live video programs in China. In broadcasting the national Spring Festival Show 2007, TrustStream attracted around 42, 850 simultaneous views and produced a peak server bandwidth of 6.70 Gbps. An increase of 160% user online viewing time justifies users' satisfaction with the quality of service provided by our live streaming service. The success of TrustStream in the real Internet demonstrates the advantage of our system and the effectiveness of a PSP structure for large-scale streaming.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments.

References

[1] C. Abad, I. Gupta, and W. Yurcik, "Adding confidentiality to application-level multicast by leveraging the muticast overlay," in *Proc. IEEE 4th Int. Workshop Assurance Distributed Syst. Netw. (ADSN)*, Columbus, OH, 2005, pp. 5–11.

[2] E. Akyol, A. M. Tekalp, and M. R. Civanlar, "A flexible multiple description coding framework for adaptive peer-to-Peer video streaming," *IEEE J. Sel. Topics Signal Process.*, vol. 1, no. 2, pp. 231–245, Aug. 2007.

[3] P. Baccichet, T. Schierl, T. Wiegand, and B. Girod, "Low-delay peer-to-peer streaming using scalable video coding," in *Packet Video 2007*, Nov. 2007, pp. 173–181.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommeareddy, "Scalable application layer multicast," in *Proc. ACM SIGCOMM*, Aug. 2002, pp. 205–217.

[5] A. Biliris, C. Cranor, F. Douglis, M. Rabinovich, S. Sibal, O. Spastcheck, and W. Sturm, "CDN brokering," *J. Comput. Commun.*, pp. 393–402, Mar. 2002.

[6] G. Blakowski and R. Steinmetz, "A media synchronization survey: Reference model, specification, and case studies," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 1, pp. 5–35, Jan. 1996.

[7] I. Chang, R. Engel, D. Kandlur, D. Pendarkis, and D. Saha, "Key management for secure Internet multicast using Boolean function minimization techniques," in *Proc. IEEE INFOCOM 1999*, pp. 689–698.

[8] K. Chan and S. H. G. Chan, "Key management approaches to offer data confidentiality for secure multicast," *IEEE Trans. Netw.*, vol. 17, no. 1, pp. 30–39, Sep./Oct. 2003.

[9] K. T. Chen, C. Y. Huang, P. Huang, and C. L. Lei, "Quantifying skype user satisfaction," in *Proc. ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communicat (SIGCOMM 2006)*, Pisa, Italy, Sep. 2006, pp. 399–410.

[10] Y. Chu, S. Rao, and H. Zhang, "A case for end system multicast," in *Proc. ACM Sigmetrics*, Santa Clara, CA, Jun. 2000, pp. 1–12.

[11] 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, pp. 139–149, Feb. 2003.

[12] A. Ganjam and H. Zhang, "Internet multicast video delivery," *Proc. IEEE*, vol. 93, no. 1, pp. 159–170, Jan. 2005.

[13] X. J. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross, "Insight into PPLive: A measurement study of a large-scalep IPTV system," in *Proc. WWW 2006 Workshop of IPTV Services Over World Wide Web*, UK, May 2006, p. 2.

[14] A. M. Kermarrec, L. Massoulie, and A. J. Ganesh, "Probabilistic reliable dissemination in large-scale systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, no. 3, pp. 248–258, Mar. 2003.

[15] J. Kim, R. M. Mersereau, and Y. Altunbasak, "Distributed video streaming using multiple description coding and unequal error protection," *IEEE Trans. Image Process.*, vol. 14, pp. 849–861, 2005.

[16] B. Li and H. Yin, "The peer-to-peer live video streaming in the Internet: Issues, existing approaches and challenges," *IEEE Commun. Mag.*, vol. 45, no. 6, pp. 94–99, Jun. 2007.

[17] J. Li, "Peerstreaming: An on-demand peer-to-peer media streaming solution based on a receiver-driven streaming protocol," in *Proc. IEEE Int. Workshop Multimedia Signal Processing*, Oct. 2005, pp. 197–200.

[18] S. Li, F. Wu, and Y.-Q. Zhang, "Experimental Results With Progressive Fine Granularity Scalable (PFGS) Coding," ISO/IEC JTC1/SC29/WG11 , MPEG99/m5742, Mar. 2000.

[19] Limelight Networks [Online]. Available: http://www.limelightnetworks.comW. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 11, no. 3, Mar. 2001

[20] M. J. Lin and K. Marzullo, "Directional gossip: Gossip in a wide-area network," Comp. Sci. Eng. Dept., Univ. California, San Diego, 1999, Tech. Rep. CS1999-0622.

[21] M. T. Lu, J. C. Wu, K. J. Peng, P. Huang, J. J. Yao, and H. H. Chen, "Design and evaluation of a P2P IPTV system for heterogeneous networks," *IIEEE Trans. Multimedia*, vol. 9, no. 8, pp. 1568–1579, Dec. 2007.

[22] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Distributing streaming media content using cooperative networking," Microsoft Research, Redmond, WA, 2002, Tech.Rep. MSR-TR-2002-37.

[23] S. Rafaeli and D. Hutchison, “A survey of key management for secure group communication,” *ACM Comput. Surv.*, vol. 35, no. 3, pp. 309–329, Sep. 2003.

[24] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network,” in *Proc. ACM SIGCOMM*, Aug. 2001, pp. 161–172.

[25] Y. Tu, J. Sun, M. Hefeeda, and S. Prabhakar, “An analytical study of peer-to-peer media streaming systems,” *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 1, no. 4, pp. 354–376, Nov. 2005.

[26] S. A. Theotokis and D. Spinellis, “A survey of P2P content distributing technologies,” *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, Dec. 2004.

[27] V. Venkataraman, K. Yoshida, and P. Francis, “Chunkyspread: Heterogeneous unstructured tree-based peer-to-Peer multicast,” in *Proc. 14th IEEE Int. Conf. Network Protocols*, Nov. 2006, pp. 2–11.

[28] C. K. Wong, M. G. Gouda, and S. S. Lam, “Secure group communications using key graphs,” *IEEE/ACM Trans. Netw.*, vol. 8, no. 1, pp. 16–30, Jan. 2000.

[29] D. Wu, Y. Hou, W. Zhu, Y.-Q. Zhang, and J. Peha, “Streaming video over the Internet: Approaches and directions,” *IEEE Trans. Circuits Systems Video Technol.*, vol. 11, no. 3, pp. 282–300, Mar. 2001.

[30] D. Wu, T. Hou, and Y.-Q. Zhang, “Transporting real-time video over the Internet: Challenges and approaches,” *Proc. IEEE*, vol. 88, no. 12, pp. 1855–1875, Dec. 2000.

[31] F. Wu, S. Li, and Y.-Q. Zhang, “A framework for efficient progressive fine granularity scalable video coding,” *IEEE Trans.Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 282–300, Mar. 2001.

[32] D. Xu, C. Rosenberg, S. Kulkarni, and H.-K. Chai, “Analysis of a CDN-P2P hybrid architecture for cost-effective streaming distribution,” *ACM/Springer Multimedia Syst. J.*, vol. 11, no. 4, pp. 585–599, 2006.

[33] H. Yin, C. Lin, F. Qiu, J. Liu, and B. Li, “CASM: A content-aware protocol for secure video multicast,” *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 270–277, Apr. 2006.

[34] H. Yin, F. Qiu, C. Lin, G. Min, and X. Chu, “A novel key-embedded scheme for secure video multicast systems,” *Int. J. Comput. Electr. Eng.*, vol. 32, pp. 376–393, 2006.

[35] L. Zhao, J. G. Luo, M. Zhang, W. J. Fu, J. Luo, Y. F. Zhang, and S. Q. Yang, “Gridmedia: A practical peer-to-peer based live video streaming syst,” in *Proc. IEEE Int. Workshop Multimedia Signal Processing*, Oct. 2005, pp. 201–204.

[36] X. Zhang, J. Liu, and B. Li, “On large scale peer-to-peer live video distribution: Coolstreaming and its preliminary experimental results,” in *Proc. IEEE Int. Workshop Multimedia Signal Processing*, Oct. 2005, pp. 185–188.

Hao Yin received the B.S., M.E., and Ph.D. degrees from Huazhong University of Science and Technology, China, in 1996, 1999, and 2002, respectively, all in electrical engineering.

Since 2003, he has been with the Department of Computer Science, Tsinghua University, Beijing, China, where he is currently an Associate Professor. His research interests span broad aspects of network architecture, P2P technology, wireless network, video coding, multimedia communication over wireless network, and security. He has published over 50 papers in refereed journal and conferences.

Dr. Yin is on editorial boards of *Advances in Multimedia* and *Ad Hoc Networks Journal.* In addition, he has been involved in organizing over 12 conferences.

Chuang Lin received the B.S. degree in 1977 and the M.S. degree from the Graduate School of the Chinese Academy of Sciences, Beijing, China, in 1981, and the Ph.D. degree in computer science from Tsinghua University, Beijing, China, in 1994.

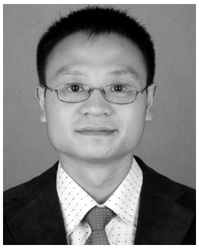
He is a Professor at the Department of Computer Science and Technology, Tsinghua University, China. He served as the Dean of Computer Science Department at Tsinghua University during 2004 and 2007. His current research interests include computer networks, performance evaluation, logic reasoning, and Petri net theory and its applications. He has co-authored more than 150 papers in research journals and IEEE conference proceedings in these areas and has published three books.

Dr. Lin is on editorial boards of *Computer Networks*, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, etc., and is the Duty Director for Internet Technical committee, Network and Data Communication Technical committee,and Petri Net Technical committee of China Computer Federation, and the Chinese Delegate in TC6 of IFIP.

Qian Zhang received the B.S., M.S., and Ph.D. degrees from Wuhan University, Wuhan, China, in 1994, 1996, and 1999, respectively, all in computer science.

She joined the Hong Kong University of Science and Technology, Hong Kong, in September 2005 as an Associate Professor. Before that, she was at Microsoft Research Asia, Beijing, China, since July 1999, where she was the Research Manager of the Wireless and Networking Group. She has published about 150 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of about 30 pending patents. Her current research interests are in the areas of wireless communications, IP networking, multimedia, P2P overlay, and wireless security.

Dr. Zhang has received the TR 100 (MIT Technology Review) world’s top young innovator award.



Zhijia Chen received the B.S. degree from Harbin Institute of Technology, China, in 2005, and is currently pursuing the Ph.D. degree in the Department of Computer Science and Technology, Tsinghua University, Beijing, China.

He was a Visiting Student at the School of Engineering of Stanford University in Spring 2007 and an exchange student at the Computer Science Department of Hong Kong University of Science and Technology in 2004. His research area is in computer network and media streaming, especially for architecture design and analytical model for P2P media streaming system.

Dapeng Wu (S’98–M’04–SM’06) received the Ph.D. in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2003.

Since August 2003, he has been with Electrical and Computer Engineering Department at University of Florida, Gainesville, as an Assistant Professor. His research interests are in the areas of networking, communications, multimedia, signal processing, and information and network security.

Dr. Wu received the NSF CAREER award in 2007, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Best Paper Award for Year 2001, and the Best Paper Award in International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine) 2006. Currently, he serves as the Editor-in-Chief of *Journal of Advances in Multimedia*, and an Associate Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and *International Journal of Ad Hoc and Ubiquitous Computing*.