Cooperation Stimulation Strategies for Peer-to-Peer Wireless Live Video-Sharing Social Networks

W. Sabrina Lin, *Member, IEEE*, H. Vicky Zhao, *Member, IEEE*, and K. J. Ray Liu, *Fellow, IEEE*

***Abstract*—Human behavior analysis in video sharing social networks is an emerging research area, which analyzes the behavior of users who share multimedia content and investigates the impact of human dynamics on video sharing systems. Users watching live streaming in the same wireless network share the same limited bandwidth of backbone connection to the Internet, thus, they might want to cooperate with each other to obtain better video quality. These users form a wireless live-streaming social network. Every user wishes to watch video with high quality while paying as little as possible cost to help others. This paper focuses on providing incentives for user cooperation. We propose a game-theoretic framework to model user behavior and to analyze the optimal strategies for user cooperation simulation in wireless live streaming. We first analyze the Pareto optimality and the time-sensitive bargaining equilibrium of the two-person game. We then extend the solution to the multiuser scenario. We also consider potential selfish users' cheating behavior and malicious users' attacking behavior and analyze the performance of the proposed strategies with the existence of cheating users and malicious attackers. Both our analytical and simulation results show that the proposed strategies can effectively stimulate user cooperation, achieve cheat free and attack resistance, and help provide reliable services for wireless live streaming applications.**

***Index Terms*—Game theory, multimedia social network, peer-to-peer video streaming, wireless network.**

I. Introduction

With the explosive advance of communication technologies and multimedia signal processing, over millions of users share multimedia data over Internet. These users interact with each other and form large-scale video-sharing social networks. Users influence each others' decisions and performance in those large scale social networks. Due to the large number of users, the behavior dynamics among users is very complex, and it raises a critical issue to formulate the complex user dynamics and analyze the impact of human factors on multimedia systems. Such investigation provides fundamental guidelines to the design of secure and personalized services.

Manuscript received June 16, 2010; revised January 23, 2010. First published March 11, 2010; current version published June 16, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jeng-Neng Hwang.

W. S. Lin and K. J. R. Liu are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: wylin@umd.edu; kjrliu@umd.edu).

H. V. Zhao is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4 Canada (e-mail: vzhao@ece.ualberta.ca).

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/TIP.2010.2045035

A peer-to-peer live streaming network [1] is one of the biggest multimedia social networks on the internet, consisting of a server broadcasting live streams and users all over the Internet watching the live program simultaneously. Recent development on wireless local area network (WLAN) enable users to utilize WLAN with low cost and high quality of service [2]–[5]. WLANs are becoming rapidly ingrained in our daily lives via public hotspots, access points, and digital home networks. Live-streaming users in the same WLAN form a wireless live-streaming social network. Due to the instability of wireless channels, cooperation among users is even more important than live-streaming networks in wired networks.

Users in the wireless live-streaming social network may use different types of devices, for example, laptops, PDAs, cellphones, and mp3/video players. Different tusers have different requirement on quality and power. For instance, laptop users would prefer higher resolution of videos, and they are willing to use more transmission power for cooperation than PDA users. Game theory [6] is a powerful tool to model the interaction among users, and to analyze the optimal cooperation strategies. In this paper, we focus on providing incentives for users to cooperate with each other. The cooperation strategy must stimulate cooperation as well as be cheat-proof and attack-resistant to ensure fairness and security of the system. In a wireless live-streaming system, all members directly download the video chunks from the server in the Internet. However, all users share the same link through the access point to the Internet and each user has different playback time and ask for different chunks at the same time. Also there are other users in the wireless network accessing Internet simultaneously. Thus, the link might be busy and some chunks can not be received by the end users in time for the playback time. Furthermore, many of the users in the wireless networks have high mobility. Therefore, they would change physical positions from time to time and the quality of network connections may be unstable. All these factors motivate user stimulation in wireless live-streaming social networks to cooperate with each other.

In the literature, the work in [7] proposed an auction-based mechanism for wireless peer-to-peer (P2P) file sharing, and the work in [8] studied the capacity of user-cooperation in wireless network. Game theory modelling of the interactions of peers in a wired peer-to-peer network have been studied in [9]–[11].

For peer-to-peer video live sharing, a rank-based peer-selection mechanism was introduced in [12], where each user is ranked by a score. A peer with a higher score has more flexibility in peer selection and, thus, receives better-quality videos. In [13], a payment-based incentive mechanism was proposed, where peers pay points to receive data and earn points by forwarding data to others. They bid for resources on

their desired data suppliers using a first-price auction procedure. Both [12], [13] use reputation or micro-payment based mechanisms, which often demand a centralized architecture, thus, hinder their scalability. The work in [14] proposed a distributed incentive mechanisms on mesh-pull P2P live video streaming systems, using layer video coding with a tit-for-tat strategy. However, there is no prior work on user cooperation stimulation and addressing cheating behavior of selfish users in wireless live video-sharing social networks. In real-world social networks, there are always users with different objectives and everyone wants to maximize his or her own payoff. Some users wish to achieve maximum utility by all means, and they will *cheat* to other users if they believe cheating can help improve their payoffs. In addition, there might also exist *malicious* users who aim to exhaust others' resources and attack the system. For example, they can tamper the media files with the intention of making the content useless (the so-called "pollution" attack) [15]. They can also launch the denial of service (DoS) attack to exhaust other users' resources and make the system unavailable [16]. Furthermore, since the peer-to-peer system is fully distributed without any centralized ringleaders, once an attacker is detected, he/she can leave the network and join again with different ID to cause more damage to the social networks. This "handwash attack" is very powerful in an anonymous system. Therefore, cheat prevention and attack resistance are fundamental requirements in order to achieve user cooperation and provide reliable services.

In this paper, we focus on designing cooperation stimulation strategies for wireless live streaming social networks. We use game theory as a tool to model user behavior and find the equilibrium to stimulate cooperation. We first model the cooperation between two users as a Bayesian game and investigate the Bayesian–Nash equilibria. Since this game usually has multiple equilibria, we then investigate how to apply extra optimality criteria, such as Pareto optimality, bargaining, and cheat-proofing, to further refine the obtained equilibrium solutions. Such analysis aims to stimulate each pair of users in the wireless live-streaming game to cooperate with each other and achieve better performance. Then, we address the issue of cooperation stimulation among multiple users and investigate cheat-proof and attack-resistant incentive mechanisms. We consider the pollution attack, incomplete-chunk attack, and handwash attack in our model. Our proposed cheat-proof and attack-resistant mechanism rewards users who contribute more with more video chunks (and, thus, better quality). It includes a request-answering algorithm for the data supplier to upload more to the peers from whom he/she downloads more, and a chunk-request algorithm for the requestor to address the tradeoffs among different quality measure and to optimize the reconstructed video quality.

The rest of this paper is organized as follows. Section II introduces the wireless live-streaming system model and the two-player game-theoretical framework. Section III studies the two-player game and the equilibria. In Section IV, a cheat-proof and attack-resistant strategy is proposed to stimulate user cooperation among all users in P2P wireless live streaming. Two more issues of wireless live video-sharing, multiple-layered coding and broadcasting nature of wireless channels, are discussed in Section V, and the final wireless live video-sharing cooperation strategy that incorporate these two issues is also studied. Section VI shows simulation results to evaluate the performance of the proposed strategies. Finally, Section VII concludes this paper.

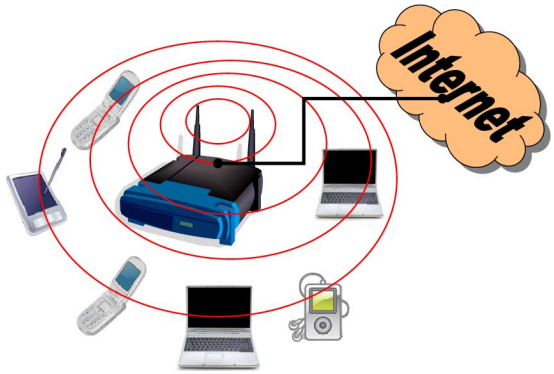


Fig. 1. Illustration of a wireless live-streaming social network.

II. System Model and Two-Player Game

In this section, we first describe the model of wireless live streaming systems and how two users in a wireless live streaming social network cooperate with each other. We then define the payoff function and introduce the game-theoretic framework of user dynamics.

A. Wireless Live Streaming Model

Fig. 2 shows the architecture of a wireless video live-streaming social network. The wireless network service is provided by an access point connected to the Internet. The video bit stream is divided into media chunks of M' bits in the original server, and are channel-coded to M bits, which is equivalent to t-second piece. All chunks are available at the streaming server in the Internet. Here we assume that there is a dedicated channel of bandwidth BHz for user cooperation and this channel is different from the channel between users and the access point. We assume that the channel for cooperation between users is symmetric and is a slow fading channel with additive white Gaussian noise with variance σ_n^2. Here we adopt the wireless signal model in [17]

$$Y_i = Z_i + \frac{A_{ij}}{\sqrt{d_{ij}}} X_i \tag{1}$$

where X_i is the signal transmitted to user i, Y_i is the signal that user i receives, Z_i is the additive Gaussian noise, A_{ij} is the channel fading factor, and d_{ij} is the distance between user i and user j.

We assume that two users, u_1 and u_2 try to cooperate with each other by exchanging chunks. Each user has a buffer of length L, which keeps L_f chunks to be played, and $L - L_f$ chunks that have been played. First u_1 and u_2 exchange information about the availability of each chunk in the other's buffer, and the transmission power P_1 and P_2 that u_1 and u_2 use to transmit the chunks, respectively. To ensure quality of cooperation, intuitively, users will not cooperate with people who use too small power for cooperation. Hence, we assume that P_1 and

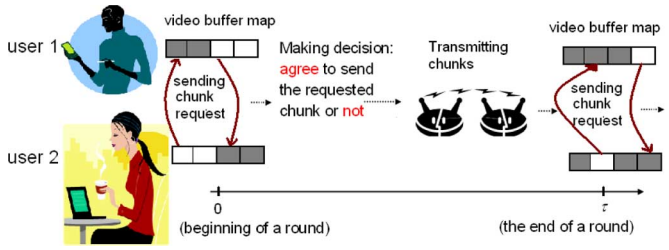


Fig. 2. Cooperation model for users in the P2P live streaming social network.

P_2 are larger than the minimum transmission power required $P_{\min}$. The users have to agree on transmitting using power larger than $P_{\min}$ before cooperation. However, the users might want to cheat on the transmit power to pay less cost. We will discuss the cheating behavior and how to prevent cheating in the following sections. The chunk exchange is done on a round by round basis. At the beginning of each round, each user sends requests to the other users, and at the same time keeps downloading from the original server. Each user is allowed to send multiple requests in each round, and he/she can also answer multiple requests. Let τ be the duration of each round. Fig. 2 shows how two users cooperate with each other: At the beginning of each round, every user sends chunk requests to each other. Then, the supplier either replies with the requested chunks and starts transmission or rejects the request. After a round duration τ, the same request-answering process is repeated.

B. Two-Player Game Model

To simplify the analysis, we start with modeling the cooperation in each round as a two-person game with single-layer video coding structure. Note that in a mesh-pull live streaming system, although all users watch the same real-time video, the progress at video playback on a peer is determined by how fast the peer collects video chunks from the system. When a new user enters the network, before starting playing the video, he/she waits for a while until he/she has received the first few chunks in the sequence and has buffered enough continuous chunks. Therefore, due to the diverse network conditions and the fact that chunks may arrive out of order, variations in chunk retrieval time result in different *playback time* for different peers and introduce time lags among users. It has been shown [18] that in pplive, one of the most popular IPTV deployments, the maximum time lag among peers fluctuates around 150 s within a one hour time period. In this scenario, every chunk has the same value, thus, users will always request chunks closest to their playback time. Assume that in the original structure, every user in the wireless live-streaming social network only asks the original server in the Internet for the media chunks, and two of them, u_1 and u_2, want to see if they can cooperate with each other to get a better-quality video. We model the interactions between u_1 and u_2 using the following game:

- **Players and player types**: There are two players, u_1 and u_2, in this game. Each player u_i has a type $\theta_i \in \{\text{laptop}, \text{PDA}, \text{PDA2}\}$. Users with different types will have different cost of sharing chunks and gain of obtaining chunks. We assume PDA2 carries weaker battery than PDA, thus, the cost per unit energy for PDA2 is higher than PDA.
- **Strategies**: In each round, the two players first exchange their buffer information, and then send the chunk requests to each other. Upon receiving the chunk requests, each player u_i decides how many chunks he/she will send to the other user in this round. We define the number of chunks u_i agrees to send as his/her strategy $a_i \in \mathbb{Z}$. Note that the two users are using the same channel, so the bits to be transmitted within a round can not be larger than the channel capacity, which equals $B \times \log(\text{SNR}+1)$. Therefore, the constraint of strategy profile (a_1, a_2) at round k is

$$\frac{a_1}{\log(1+P_1A_{12}^2/d_{12}\sigma_n^2)} + \frac{a_2}{\log(1+P_2A_{21}^2/d_{12}\sigma_n^2)} \le \frac{\tau B}{M}. \tag{2}$$

  If (2) is not satisfied and the users are transmitting chunks above the channel capacity, the probability of transmission would be high and neither will receive any chunks successfully.
- **Utility function**: The utility function π_i of u_i is considered as the gain of receiving chunks (with respect to the opponent's action) minus the cost of sending chunks (his/her own action). Since the members in the wireless live-streaming social network are using mobile devices, the battery energy is the most limited resource. Hence, the cost of cooperation is considered as the transmission energy, and each type of player would give a different weight to the energy cost. For example, clients running on tight energy budget bear a higher cost than those with powerful batteries. Let c_i be the cost per unit energy for u_i, and g_i be u_i's gain of completely receiving one chunk. Every user in the P2P wireless live streaming social network defines his/her own value of g_i depending upon how much he/she wants to watch the video. For instance, assume that the NFL final is being broadcasted. An NFL fan would want to try his/her best to receive a high quality video to enjoy the game better, and he/she will set g_i to 1. Another user is watching the game and a movie at the same time. He/she is more interested in the movie, but wants to check the scores/result of the NFL game from time to time. For this user, he/she may give a higher priority to the movie channel, and uses a lower g_i for the streaming of the NFL game.

Based upon the previous discussion, given the strategy profile (a_1, a_2), the players' payoffs for the kth round are formulated as follows:

$$\begin{aligned}\pi_1(a_1,a_2) &= a_2g_1 - a_1c_1\frac{MP_1}{B\log\left(1+\frac{P_1A_{12}^2}{d_{12}\sigma_n^2}\right)}\\ \pi_2(a_1,a_2) &= a_1g_2 - a_2c_2\frac{MP_2}{B\log\left(1+\frac{P_2A_{21}^2}{d_{12}\sigma_n^2}\right)}.\end{aligned} \tag{3}$$

Let $\pi(a_1,a_2) = (\pi_1(a_1,a_2), \pi_2(a_1,a_2))$ be the payoff profile. Define $K_1 = MP_1/B\log(1+P_1A_{12}^2/d_{12}\sigma_n^2)$, and $K_2 = MP_2/B\log(1+P_2A_{12}^2/d_{12}\sigma_n^2)$. K_i can be considered as

the energy that user i spends on transmitting a chunk. It is reasonable to assume that $g_i \geq c_i K_i$ and there exists a $C_{\max}$ where $c_i K_i \leq C_{\max}$. Here c_i and g_i are user i's private information depending upon user i's type, and are not known to others. We assume that users do not exchange their private information, i.e., their types. Thus, this is a game with incomplete information. We assume that users have the belief of the probability of the other users' type, which is independent of their own type. Let p_1, p_2, and p_3 be the probability of a user being a laptop, PDA, and PDA2, respectively.

III. Optimal Strategies Analysis for Two-Player Game

In this section, we first extend the one-stage game model in Section II-B into a infinitely repeated game, then apply several optimization criteria such as Pareto optimality and time-sensitive bargaining solution to refine the Bayesian–Nash equilibriums of the game. Furthermore, we discuss the possible cheating behavior which all users may apply to increase their own utility, and design cheat-proof cooperation strategy to stimulate cooperation between two users.

A. Repeated Game Model

It is easy to show that, if the previously mentioned game will only be played for one time, the only Bayesian–Nash equilibrium is (0,0), which means no one will answer the other's requests. According to the backward induction principle [19], there will also be no cooperation between the two users when the repeated game will be played for finite times with game termination time known to both players. Therefore, in both circumstances, the only optimal strategy for both players is to always play noncooperatively.

However, in live streaming, these two players will interact many rounds and no one can know exactly when the other player will quit the game. Thus, we can model the dynamics between u_1 and u_2 as an infinitely repeated game, and we will show in the following section that cooperative strategies can be obtained in this realistic model. Let s_i denote player i's behavior strategy, and let $\mathbf{s}_1 = (s_1^{(1)}, s_1^{(2)}, \ldots, s_1^{(T)})$, $\mathbf{s}_2 = (s_2^{(1)}, s_2^{(2)}, \ldots, s_2^{(T)})$ denote the strategy profile till the Tth round. Next, we consider the following utility function of the infinitely repeated game:

$$U_i(\mathbf{s}) = \lim_{T\to\infty} \sum_{t=1}^{T} u_i(s^{(t)}). \tag{4}$$

Now, we analyze the Bayesian–Nash equilibriums for the infinitely repeated game with the previously mentioned utility function U_i. According to the Folk theorem [19], there exists at least one Bayesian–Nash equilibrium to achieve every feasible and enforceable payoff profile, where the set of feasible payoff profiles for the previously mentioned game is:

$$V_0 = \text{convex hull } \{v | \exists\ (a_1, a_2) \text{ with} \tag{5}$$
$$cr\ (\pi_1(a_1, a_2), \pi_2(a_1, a_2)) = (v_1, v_2)\}$$
$$\text{where } a_1, a_2 \text{ satisfy } (2) \tag{6}$$

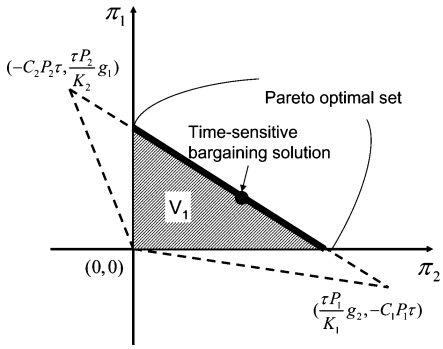


Fig. 3. Feasible and enforceable payoff profiles.

and the set of enforceable payoff, denoted by V_1, can be easily derived

$$V_1 = \{(v_1, v_2) | (v_1, v_2) \in V_0 \text{ and } v_1, v_2 \geq 0\}. \tag{7}$$

Fig. 3 illustrates both the feasible region and the enforceable region: the feasible region is inside the triangle bounded by dashed lines, and the enforceable feasible set V_1 is the shaded region shown in Fig. 3. It is clear that there exists an infinite number of Bayesian–Nash equilibriums (BNE). To simplify our equations, in this paper, we use $\mathbf{x} = (x_1, x_2)$ to denote the set of BNE strategies corresponding to the enforceable payoff profile $(x_2 g_1 - x_1 c_1 K_1, x_1 g_2 - x_2 c_2 K_2)$.

From the previously mentioned analysis, one can see that the infinitely repeated game has infinite number of equilibriums, and apparently, not all of them are simultaneously acceptable. For example, the payoff profile (0,0) is not acceptable from both players' point of view. Therefore, in this section, we will discuss how to refine the equilibriums based upon new optimality criteria to eliminate those less rational and find which equilibrium is cheat-proof.

B. Nash Equilibrium Refinement

The following optimality criteria will be considered in this section: Pareto optimality, proportional fairness, and absolute fairness.

Pareto Optimality: A payoff profile $v \in V_0$ is Pareto Optimal if and only if there is no $v' \in V_0$ that $v_i' \geq v_i$ for all $i \in N$ [6]. Pareto Optimality means no one can increase his/her payoff without degrading other's, which the rational players will always go to. It is clear from Fig. 3 that the solid segment between $(-C_2P_2\tau, g_1\tau P_2/K_2)$ and $(g_2\tau P_1/K_1, -C_1P_1\tau)$ in the first quadrant is the Pareto Optimal set.

Time-Sensitive Bargaining Solution: Since the players' action pair (a_1, a_2) has to satisfy (2), and both players are rational and greedy, they will try to maximize the quality of their live streaming by asking as many chunks as possible in each round. Every user will request all the chunks that his/her opponent has and that he/she needs. However, according to information theory, the total number of bits being transmitted in within a round has to be less than the channel capacity times chunk duration τ to ensure that the information can be transmitted without bit error. Here we adopt time division multiple access (TDMA) scheme that divide a round time into several time slot, and within

a time slot, only one user is occupying the band. Thus, users have to bargain for their chunk-request quota for every round to ensure that the total number of bits to be transmitted is not larger than the channel capacity. Also, the gain of receiving a chunk is time-sensitive. For instance, if users cannot reach an agreement on time a user has no gain by receiving that chunk after the playback time.

We model the time-sensitive process for round k as follows: one user offers an action pair $(a_1^{(1)}, a_2^{(1)})$ first, and the other user can decide whether to accept this offer or to reject and offer back another action pair $(a_1^{(2)}, a_2^{(2)})$. This process continues until both players agree on the offer. If users reach agreement at the jth action pair, then g_i decreases to $\delta_i^{j-1}(LC_{k,i})g_i$ for $i = 1$ or 2, where $\delta_i(LC_{k,i})$ is the discount factor for $u_i, LC_{k,i} = \{I_1, \ldots, I_q\}$ denotes the indexes of chunks u_i wants to ask in the kth round, and $I(k)$ denotes the index of the chunk playing at the beginning of kth round. Let t be the length of a chunk (in seconds). Suppose the first q' terms in $LC_{k,i}$ are smaller than $I(k)+\tau/t$, which means that among all the chunks that user i needs, there are q' of them have the playback time within the same (kth) round. Therefore, for these q' chunks, if users cannot reach agreement within the kth round, user i gains nothing by receiving them since their playback time has already passed. For the rest $q-q'$ chunks, which would be played after the kth round, user i still receives gain by receiving them still preserve even if bargaining process does not end within a round duration. On the other hand, if one of these $q-q'$ chunks can be received in the kth round, its value is guaranteed to be g_i. However, if the bargaining process in round k takes more time, the number of chunks that can be transmitted in the kth round would decrease. Consequently, a smaller portion of the $q-q'$ chunks can be received in the kth round, thus, users receive a small gain. Therefore, even for the chunks which would be played after the kth round, their value would have a higher risk to be dropped if the bargaining time in the kth round is longer.

According to the previously mentioned analysis, we define the discount factor of gain for user i at round k as follows:

$$\delta_i(LC_{k,i}) = 1 - \frac{\sum_{i=1}^{q'} \frac{\tau}{t} - (I_i - I(k)) + (q-q') * d}{\frac{\tau}{t}(\frac{\tau}{t}+1)/2 + (L_f - \frac{\tau}{t}) * d} \tag{8}$$

where $\mathrm{d} < 1$ is the discount constant of the chunks that will be played after the $k+1$th round begins. For the $q-q'$ chunks that are scheduled to be played after the end of the kth round, it is also better to receive them as soon as possible to prevent their value becomes zero. From such aspect, the value of these $q-q'$ chunks is also decreasing with time and should be counted in δ. However, the value of $q-q'$ does not decrease as fast as the q' chunks that have to be played within this round, and these $q-q'$ chunks should not play equal roles as the q' chunks that have to be received within this round. So d is the factor to evaluate the less-importance of these $q-q'$ chunks.

For each of the q' chunks whose playback time is within the kth round, the later its playback time, the higher chance that the gain of receiving it can be preserved. We use the chunk index difference to model this phenomena. Thus, the first term in the numerator of (8) is the sum of the index difference between the requested chunks and the last chunk that can be played in the kth round.

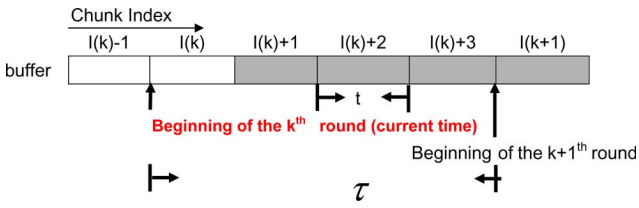


Fig. 4. Example of a user's buffer with $\mathrm{length} = 5$ chunks.

Fig. 4 gives as an example to illustrate the time-sensitive property for the live-streaming scenario: the white blocks are the chunks that u_1 has in buffer, the grey ones are the chunks he/she needs, and the buffer contains $L = L_f = 5$ chunks. In this example, the number of chunks that u_1 would request, $q = 4, q' = 3$, and $\tau/t = 4$. Therefore, $\sum_{i=1}^{q'} \frac{\tau}{t} - (I_i - I(k)) = (4-1) + (4-2) + (4-3) = 6$, and $q - q' = 1$. Let $d = 0.8$, then the discount factor of gain for user i at round $k, \delta_i(LC_{k,i}) = 0.37$.

Since both players' payoffs decrease as the time for bargaining increases, the first mover would seek the equilibrium and offer at the first bargaining round for his/her maximum payoff. Let δ_1 and δ_2 be the averaged discount factor for u_1 and u_2 over all rounds. Note that here we are discussing about the equilibrium of the infinite game, which is the outcome when the game goes to infinity. So at each round, the users do not need to predict δ_i that is averaged over all rounds (including the future). Instead, for each round, the users can calculate the averaged δ_i till the previous round, and find the equilibrium. Such mechanism will result in the equilibrium as follows: The Pareto-optimal equilibrium pair $((x_1^{(1)}, x_2^{(1)}), (x_1^{(2)}, x_2^{(2)}))$ for the infinitely repeated game happens when

$$\begin{aligned} & x_2^{(2)} g_1 - x_1^{(2)} c_1 K_1 = \delta_1 x_2^{(1)} g_1 - x_1^{(1)} c_1 K_1 \\ & x_1^{(1)} g_2 - x_2^{(1)} c_2 K_2 = \delta_2 x_1^{(2)} g_2 - x_2^{(2)} c_2 K_2 \\ \text{where} \quad & x_1 \frac{K_1}{P_1} + x_2 \frac{K_2}{P_2} = \tau. \end{aligned} \tag{9}$$

Since two users take turn to make the first offer, the time-sensitive bargaining strategy (x_1^*, x_2^*) is

$$\begin{aligned} x_1 &= \frac{1+m}{2} \times \frac{(1-\delta_1)\frac{P_2}{K_2} g_1 \tau}{(m-1)K_1 c_1 + (m-\delta_1)\frac{K_1 P_2}{K_2 P_1} g_1} \\ x_2 &= P_2 \frac{\tau - x_1 \frac{K_1}{P_1}}{K_2}, \text{ where } m = \frac{g_2 + c_2 K_2 \frac{P_2}{P_1}}{\delta_2 g_2 + c_2 K_2 \frac{P_2}{P_1}}. \end{aligned} \tag{10}$$

It is clear that the bargaining solution in (10) depends upon the knowledge of both users' types, i.e., the private information, which is unavailable. Both players know the discount factors δ_1, δ_2 since the discount factors only depend upon the chunks to be requested, which is the information the two users have to exchange. Although at the beginning, users do not know each other's type, they can probe it during the bargaining process using the following mechanism: Let T_1 be u_1's type, which is only known to u_1, let T_2 be u_2's type

and $T(j)$ is the jth type. At the first bargaining stage, without loss of generality, let u_1 be the first mover. u_1 calculates all the bargaining equilibriums $(a_1^{(1)}(T_1,T(j)),a_2^{(1)}(T_1,T(j)))$ for $j=1,2,3$ corresponding to the three possible types of u_2. Then u_1 chooses the the equilibrium j' that gives highest $p_{j'}\pi_1(a_1^{(1)}(T_1,T(j')),a_2^{(1)}(T_1,T(j')))$. u_2 will accept the offer if $\pi_2\left(a_1^{(1)}(T_1,T(j)\right),a_2^{(1)}(T_1,T(j)))$ is larger than or equal to $\pi_2(a_1^{(1)}(T_1,T_2),a_2^{(1)}(T_1,T_2))$. If not, u_2 will offer back $(a_1^{(2)}(T_1,T_2),a_2^{(2)}(T_1,T_2))$ and reach the agreement. Since u_1 calculates the offer based upon the equilibrium in (10), which depends upon u_1's own type, u_2 can probe u_1's type based upon the offer he/she made. Thus, after the first bargaining stage in the first chunk-requesting round, u_2 knows u_1's type, and since u_2 will make the first move in next round, after 2 rounds, the both users have the information of each other's type.

C. Cheat-Proof Cooperation Strategy

Users in peer-to-peer wireless live streaming social networks would try to maximize their own utility even by cheating. Therefore, to ensure fairness and to give incentives to users, it is crucial that the cooperation strategy is cheat-proof. In this subsection, we will first discuss possible cheating methods, and then propose the two-person cheat-proof cooperation strategy in peer-to-peer wireless live streaming social networks.

1) Cheat on Private Information: Since users know each other's private information (g_i,c_i) by the offers they made, users can cheat by making different offers. First, let us exam whether the time-sensitive bargaining solution in (10) is cheat-proof with respect to (g_i,c_i): π_2 increases when x_2 decreases, which can be achieved by increasing x_1 or decreasing P_2.

x_1 is a function of m and

$$\frac{\partial x_1}{\partial m}=-\frac{(1+m)\left(K_1c_1+\frac{K_1P_2}{K_2P_1}g_1\right)(1-\delta_1)\frac{P_2}{K_2}g_1\tau}{2\left[(m-1)K_1c_1+(m-\delta_1)\frac{K_1P_2}{K_2P_1}g_1\right]^2} \tag{11}$$

which is always less than 0 since $m\geq 1\geq\delta_1$. Thus, x_1 is a monotonically decreasing function of m if $\delta_1<1$.

Furthermore

$$\frac{\partial m}{\partial g_2}=\frac{(\delta_2-1)c_2K_2\frac{P_2}{P_1}}{\left(\delta_2g_2+c_2K_2\frac{P_2}{P_1}\right)^2}\leq 0 \text{ and}$$
$$\frac{\partial m}{\partial c_2}=\frac{(1-\delta_2)K_2\frac{P_2}{P_1}}{\left(\delta_2g_2+c_2K_2\frac{P_2}{P_1}\right)^2}\geq 0. \tag{12}$$

Therefore, m is a monotonically decreasing function of g_2 and is a monotonically increasing function of c_2 if $\delta_2<0$. Thus, u_2 can have a higher payoff by making the bargain offer using lower g_2, higher c_2, and lower P_2. Similarly, u_1 can also achieve higher utility by offering the equilibrium based upon lower g_1, higher c_1, and lower P_1.

As the consequence that both players cheat with respect to c_i and g_i, from the previously mentioned analysis, both players will bargain based upon the minimum value of g_i and maximum value of c_i. Since we have assumed that $g_i\geq c_iK_i$, and $P_i\geq P_{\min}$, both players will make the offer based upon $g_i=c_iK_i=C_{\max}$, and $P_i=P_{\min}$, thus, the solution (10) becomes

$$x_1^*=\frac{(\delta_2+3)(1-\delta_1)}{2(4-(1+\delta_1)(1+\delta_2))}\times\frac{\tau}{M/B\log\left(1+\frac{P_1A_{12}^2}{d_{12}\sigma_n^2}\right)}$$
$$x_2^*=\frac{\tau}{M/B\log\left(1+\frac{P_1A_{21}^2}{d_{12}\sigma_n^2}\right)}-x_1^* \tag{13}$$

which implies that both players should always cooperate with each other. It is clear that solution in (13) forms an Nash Equilibrium, is Pareto-Optimal, and is cheat-proof with respect to private information g_i and c_i. Note that the user whose discount factor is closer to 1 has an advantage, and if $\delta_1=\delta_2$, then $x_1^*=x_2^*=$ half number of chunks can be transmitted in τ seconds.

2) Cheat on Buffer Information: The other way of cheating is to cheat on buffer information, that is, although player i has chunk k in the buffer, he/she does not report it to its opponent. In order to reduce the number of requests from its opponent. However, hiding the chunk that the other user needs might increase the other user's discount factor based upon (8).

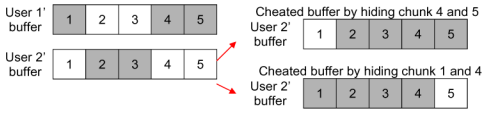


Fig. 5. Example of how to cheat on buffer information.

Take Fig. 5 as an example. The white blocks are the chunks in buffer, while the grey blocks are the chunks that the user needs. Suppose user 1 always reports his/her buffer information honestly and the time-sensitive bargaining solution gives two chunk-request quota for user 1, and two chunk-request quota for user 2. Apparently, user 1 will ask two of chunk 1, 4, 5 from user 2, and user 2 will ask chunk 2, 3 from user 1. Now if user 2 wants to hide chunks in his/her buffer from user 1, so that the number of chunk requests user 1 will send to user 2 will decrease, and increase user 2's payoff in this round. It is clear that user 2 has to hide at least 2 chunks to increase his/her payoff, since if user 2 only hides one chunk, there are still two chunks in user 2's buffer that user 1 needs. User 2 can choose two of chunk 1, 4, and 5 to hide, and hiding different chunk will lead to different utility. For instance, if user 2 hides chunk 1 and 4, which means chunk 5 is the only chunk that user 1 needs. However, user 2 would ask chunk 2 and 3 from user 1. Since chunk 4 has a later playback time than that of chunk 2 and 3, the discount factor of user 1's gain will be larger than user 2. Thus, user 1 will have more advantage in the time-sensitive bargaining process, and the bargaining solution might be changed to 3 chunk-request quota for user 1 and 1 chunk-request quota for user 2. As a result, user 2's utility decreases because now he/she can only ask one chunk from user 1. Therefore, user 2 has no incentive to cheat on buffer information by hiding chunk 1 and 4.

Although user 2's cheating on buffer information will always increase the the discount factor of user 1's gain (δ_1), it does not necessarily lead to the decrease of chunk-request quota. The

reason is the chunk-request quota is always an integer since partial chunk gives no gain for each user and the users would like to round the time-sensitive solution to the closest integers. For instance, if before cheating, the time-sensitive bargaining solution is (1.8, 2.2), and the solution changes to (2.4, 1.6) after cheating. Both solutions round to (2, 2), which means if user 2 hides the chunks properly to keep δ_1 low so that the chunk-request quota does not change after cheating, cheating on buffer information will increase user 2's utility since user 2 can still ask two chunks from user 1, and there is only one chunk in user 2's buffer that user 1 needs.

Therefore, to prevent selfish users gain higher utility by cheating on buffer information, each player should not send chunks more than the other one has sent.

3) Cheat on Transmitted Power: The power that user 1 and user 2 use for cooperation, P_1 and P_2, are declared in the beginning of the game, and they directly influence the feasible region as in Fig. 3 and the bargaining solution (13). As discussed in Section III-C-1, user i can increase his/her payoff by decreasing P_i, thus, both users will declare that they use the minimum power $P_{\min}$. However, if the user declares that he/she transmit the chunks using $P_{\min}$ but the actual power used for transmission is less than $P_{\min}$, he/she can have higher utility by paying less cost for cooperation.

Given the signal model in (1), the receiver has to estimate the attenuation term $A_{ij}/\sqrt{d_{ij}}$ before estimating the transmitted power. Suppose user i wants to estimate $A_{ij}/\sqrt{d_{ij}}$. If user j is honest, user i can simply ask user j to transmit a probing signal using P_2 to estimate the attenuation. However, in the fully distributed system, user j might be cheating and transmit the probing signal with power lower than P_2, and the estimated attenuation that user i estimated will be more serious than the real attenuation. To solve this problem, we propose that user i sends the probing signal that user j cannot decode to user j and ask user j to transmit back the received signal, and user i can investigate the attenuation from the replied signal.

If user i send the probing signal X to user j, then the signal Y_j that user j receives is $Z_j + A_{ij}/\sqrt{d_{ij}}X$. Suppose the selfish user j wants to manipulate the signal, he/she can secretly amplify Y_j with a constant $\alpha < 1$ and then send αY_j back to user i. Then the replied signal Y_i that user i receive will be

$$Y_i = Z_i + \alpha\frac{A_{ij}}{\sqrt{d_{ij}}}Z_j + \alpha\frac{A_{ij}^2}{d_{ij}}X. \tag{14}$$

Since user i knows X and the noise power σ_n^2, he/she can easily extract $\alpha(A_{ij}^2)/(d_{ij})X$ from Y_i, divide the energy of the residue with σ_n^2, and get the estimation of $1+\alpha^2$. Given α, the attenuation term $(A_{ij}^2)/(d_{ij})$ can be estimated easily. From the previously mentioned analysis, such probing procedure is cheat-proof since no matter how user j manipulates the signal, the estimation of the attenuation term is independent of α.

After estimating $(A_{ij}^2)/(d_{ij})$, the transmitted power can be easily to be estimated by calculating the averaged power of the signal at the receiver's side. Therefore, for user i, he/she can compute the estimated transmitted power $P_j'(k)$ for user j at the kth round by

$$P_j'(k) = \frac{d_{ij}}{A_{ij}^2}\frac{1}{\tau_j}\int_{t=t_k}^{t=t_k+\tau_j}\left[y^2(t) - \sigma_n^2\right] \tag{15}$$

where $y(t)$ is the received signal, t_k is the beginning of user j's transmission in the kth round, and τ_j is the duration of user j's transmission in the kth round.

Thus, we design a mechanism to prevent cheating on transmitted power based upon $P_j'(k)$ in (15):

- For each user i at each round k, he/she estimates the transmitted power of the other user j by (15). If $P_j'(k)$ is less than $P_{\min}$, then at the round (the $k+1$th round), user i transmit the chunks using $P_j'(k)$. If $P_j'(k) \geq P_{\min}$, user i uses $P_{\min}$ power for cooperation.
- Each user estimates the transmitted power at every round and follow the same decision above.

Using the previously mentioned mechanism, if user i decides to cheat by transmitting chunks with power $P_i' \leq P_{\min}$, then the other user j can estimate P_i' and use P_i' to transmit the chunks for user i in the next round. Therefore, although user i increases his/her payoff in this current round, his/her payoff will be decreased in the next round, thus, the actual channel capacity is less than the users' estimation using $P_{\min}$. Therefore, the probability of successfully receiving the request chunks for both users would decrease and lead to no gain since they cannot receive the extra chunks by cooperation. Therefore, neither of the users has the incentive to cheat on transmission power if both follow the previously mentioned mechanism.

4) Two-Player Cheat-Proof Cooperation Strategy: Based upon the previously mentioned analysis, we can conclude that, in the two-player wireless live-streaming game, in order to maximize each user's own payoff and resist possible cheating behavior, for each player in each round, he/she should always agree to send the requested chunks up to the bargained chunk-requesting quota as in (10) and should not send more chunks than his/her opponent has sent to him/her. Also, each user should estimate the transmitted power of the other user in every round, and use the estimated power for transmission if it is less than $P_{\min}$. We refer to the previously mentioned strategy as *two-player cheat-proof wireless live streaming cooperation strategy*.

IV. Multiuser P2P Wireless Live Streaming Game

In this section, we first introduce the multiuser game formulation to model the behavior of all users in a peer-to-peer live streaming social network. Then we propose a cheat-proof and attack-resistant cooperation strategy for the infinitely repeated game model, and show that the cooperation strategy is a Pareto optimal and subgame perfect Nash equilibrium. We further discuss the impact of handwash attack to the system and design the strategy to against handwash attack.

First, we will try to extend the two-player cooperation strategy derived from the previous section into the multiple-user scenario.

The two-player cooperation strategy suggests that users should be fully cooperative and refuse to cooperate with a user who behaves uncooperatively before. However, transmission errors are inevitable in fading and noisy wireless channels, and the errors can cause severe troubles. For the two-player cheat-proof cooperation strategy, there exists a positive probability that one packet and a packet cannot be decoded successfully due to transmission errors and has to be retransmitted. Retransmission may cause delay, and some packets can not arrive within one round. In such scenario, the game will be terminated immediately since the two-person cheat-proof cooperation strategy asks for equal contribution between users, and the performance will be degraded drastically. Therefore, the malicious users can claim it was due to the erroneous Internet traffic and pretend to be nonmalicious. Distinguishing misbehavior caused by bit errors and packet loss from that caused by malicious intention is a challenging task.

Also, in the multiuser scenario, the repeated game model is not applicable. For example, a peer may request chunks from different peers at different time slots to maximize his/her utility. A direct consequence of such a nonrepeated model is that favors cannot be simultaneously granted. When favors cannot be granted simultaneously, players falls into the dilemma of egoism or altruism, where egoism is an intuitive choice but will stop others from giving favors. Meanwhile, altruism may not guarantee satisfactory future payback, especially when future is unpredictable. Hence, the two-player cheat-proof and attack-resistant solution cannot be directly applied to the multiuser scenario.

A. Multiuser Game Model

Next, we will investigate how to stimulate cooperation for all users in peer-to-peer wireless live streaming over noisy channels, and analyze users' behavior dynamics. We focus on the scenario that video streaming will keep alive for a relatively long time, and there exist a finite number of users (for example, people watch live Super Bowl over the Internet). Each user will stay in the social network for a reasonably long time (for instance, from the beginning to the end of the game). They are allowed to leave and reconnect to the network when necessary. Each user has an unique user ID registered at the first time he/she joins this network for identification purpose, and he/she uses the same ID whenever he/she reconnects to the same network. We consider an information-pull model, where the streaming server has no duty to guarantee the successful delivery of chunks and it only sends out chunks upon users' demand.

For each user, uploading chunks to other users will incur cost, and successfully receiving chunks can improve the quality of his/her video and, thus, brings some gain. To simplify the analysis, in this section, we assume that the video stream is encoded using a nonscalable video codec. Therefore, for each user i, each received chunk gives the same gain g_i, whose value is specified by the user individually and independently. As discussed in Section III, g_i, the gain of receiving a chunk for the live video, is evaluated by user i depending upon how much he/she wants to watch the video.

In a real-world social network, some users may be malicious, whose goal is to cause damages to other users. In this paper, we focus on inside attackers, that is, the attackers also have legitimate identities, and their goal is to prevent selfish users from getting chunks. In P2P wireless live streaming social networks, there are three ways to attack the system:

1) **Handwash Attack**: Since peer-to-peer system has a pure anonymous nature that each user is identified by the ID they registered, if a malicious user is detected and cannot cause damage to the system anymore, he/she can delete his/her ID and register for a new one to come back to the social network. By handwashing, the attacker can keep causing damages to the system as a new comer.
2) **Incomplete chunk attack**: A malicious user agrees to send the entire requested chunk to the peer, but sends only portions of it or no data at all. By doing so, the requesting user wastes his/her request quota in this round, and has to request the same chunk again in the next round.
3) **Pollution attack**: The other kind of attack in peer-to-peer wireless live streaming is pollution [15]. In P2P wireless streaming system, a malicious user corrupts the data chunks, renders the content unusable, and then makes this polluted content available for sharing with other peers. Unable to distinguish polluted chunks from unpolluted files, unsuspecting users download the polluted chunks into their own buffers, from which others may then download the polluted data. In this manner, polluted data chunks spread through the system.

Instead of forcing all users to act fully cooperatively, our goal is to stimulate cooperation among selfish (nonmalicious) users as much as possible and minimize the damages caused by malicious users. In general, not all cooperation decisions can be perfectly executed. For example, when a user decides to send another peer the requested chunks, packets of the chunk may not be correctly decoded at the receiver's side. In this paper, we assume that the requesting peer gives up the chunk if it does not arrive in one round, and we use P_{ij} to denote the probability of successful transmission of a chunk from peer i to peer j in one round of τ second. At the beginning of every round, each user will first bargain the chunk-request quota, and then send chunk requests to others. We assume that every chunk request can be received immediately and perfectly since chunk request is very small in terms of number of bits, even if they are not received perfectly, the retransmission can be done in a very short time.

As an extension of the two-person time-sensitive bargaining in Section III-B, when there are N users bargaining for the chunk request quota, the bargaining procedure is as follows: one user offers an action N-tuple $(a_1^{(1)}, a_2^{(1)}, \ldots, a_N^{(1)})$ first, and the second user can decide whether to accept this offer or to reject and offer to the first user another action N-tuple $(a_1^{(2)}, a_2^{(2)}, \ldots, a_N^{(2)})$. If the second user agree on the offer, the third user can decide to accept or reject and offer back. And the rest of the users can also make their choices sequentially. This process continues until all players agree on the offer. If users reach agreement at the jth action pair, then g_i decreases to $\delta_i^{j-1} g_i$ for

$i \in N$. Therefore, the Pareto-optimal equilibrium N-tuple $((x_1^{(1)},\ldots,x_N^{(1)}),(x_1^{(2)},\ldots,x_N^{(2)}),\ldots,(x_1^{(N)},\ldots,x_N^{(N)}))$ will satisfy (16)

$$\begin{aligned}
&\delta_1^{N-1}x_1(1)g_1 - \sum_{i\neq 1} x_i^{(1)}c_1 \\
&\quad = \delta_1^{N-2}x_1(2)g_1 - \sum_{i\neq 1} x_i^{(2)}c_1 \\
&\quad = \cdots = x_1^{(N)}g_1 - \sum_{i\neq 1} x_i^{(N)}c_1 \\
&\delta_2^{N-1}x_2^{(2)}g_2 - \sum_{i\neq 2} x_i^{(2)}c_2 \\
&\quad = \delta_2^{N-2}x_2^{(3)}g_2 - \sum_{i\neq 2} x_i^{(3)}c_2 \\
&\quad = \cdots = x_2^{(1)}g_2 - \sum_{i\neq 2} x_i^{(1)}c_2 \\
&\quad \vdots \\
&\delta_N^{N-1}x_N^{(N)}g_N - \sum_{i\neq N} x_i^{(N)}c_N \\
&\quad = \delta_N^{N-2}x_N^{(1)}g_N - \sum_{i\neq N} x_i^{(1)}c_N \\
&\quad = \cdots = x_N^{(N-1)}g_N - \sum_{i\neq N} x_i^{(N-1)}c_N
\end{aligned}$$

and

$$\sum_{j=1}^{N} x_j^{(i)}\frac{K_j}{P_j} = \tau \ \forall i \in \{1,2,\ldots,N\}. \tag{16}$$

By solving the linear equations in (16), the N-user time-sensitive bargaining solution can be achieved.

In order to formally analyze cooperation and security in such peer-to-peer live streaming networks, we model the interactions among peers as the following game.

1) **Server**: The video is originally stored at the original streaming server with upload bandwidth W_s, and the server will send chunks in a round-robin fashion to its peers. All players are connected via the same access point to the Internet. This backbone connection has download bandwidth W_d.
2) **Players and player type**: There are finite number of users in the peer-to-peer wireless live streaming social network, denoted by N. Each player $i \in N$ has a type $\theta_i \in \{\text{selfish}, \text{malicious}\}$. Let N_s denote the set of all selfish players and $N_m = N\backslash N_s$ is the set including all inside attackers. A selfish(nonmalicious) user aims to maximize his/her own payoff, and may cheat to others if cheating can help increase his/her payoff. A malicious user wishes to exhaust other peers' resources and attack the system.
3) **Chunk requesting**: In each round, users bargain for chunk-request quota based upon the time-sensitive bargaining solution since the channel dedicated for user cooperation has limited bandwidth B. For each chunk-request quota, the user can send multiple chunk-request to one user. Users can use their chunk-request quota either *requests chunks from other users* or *does not request any chunks* in this round. On the other hand, since the user-cooperation channel is different from the channel between users and the access point, the users can ask the server for chunks at the same time.
4) **Request answering**: For each player, after receiving a request, it can either *accept* or *refuse* the requests.
5) **Cost**: For any player $i \in N$, uploading a chunk to player j incurs cost $c_i MP_i/B\log(1+(P_iA_{ij}^2)/(d_{ij}\sigma_n^2))$, where c_i is the user-defined cost per unit energy, P_i is the transmission power that player i uses for cooperation and $P_i \geq P_{\min}$, same as in Section III.
6) **Gain**: For each selfish user $i \in N_s$, if he/she requests a data chunk from another peer j, and if a clean copy is successfully delivered to him/her, his/her gain is g_i where $g_i > \max_j c_i \mathrm{MP_i/B}\log(1+(\mathrm{P_iA_{ij}^2})/(\mathrm{d_{ij}}\sigma_n^2))$.
7) **Utility function**: We first define the following symbols: for each player $i \in N$,
   — $Cr^{(i)}(j,t)$ is the total number of chunks that i has requested from j by time t. Here, j can be either a peer ($j \in N$) or j is the streaming server. $Cr^{(i)}(t) = \sum_{j\in\{N,\text{ source}\}} Cr^{(i)}(j,t)$ denotes the total number of chunks that i has requested by time t.
   — By time t, peer i has successfully received $Cs^{(i)}(j,t)$ chunks from peer j in time (a chunk is received in time if and only if it is received within the same round that it was requested). $Cs^{(i)}(t) = \sum_{j\in\{N,\text{source}\}} Cs^{(i)}(j,t)$ is peer i's total number of successfully received chunks by time t.
   — By time t, $C_p^{(i)}(j,t)$ is the total number of polluted chunks that peer i received from peer j. The total number of successively received unpolluted data chunks that peer i received from peer j is $Cs^{(i)}(j,t) - C_p^{(i)}(j,t)$, and each successfully received unpolluted chunk gives peer j a gain of g_i.
   — $Cu^{(i)}(j,t)$ denotes the number of chunks that i has uploaded to player j by time t. $Cu^{(i)}(t) = \sum_{j\in\{N,\text{source}\}} Cu^{(i)}(t)$. The cost of uploading each chunk is c_i for peer i.

   Let t_f be the lifetime of the peer-to-peer live streaming social network, and $T^{(i)}(t)$ denotes the total time that peer i is in the network by time t. Then, we model the player's utility as follows.
   1) For any selfish player $i \in N_s$, its utility $U_s^{(i)}(t_f)$ is defined as in (17), shown at the bottom of the next page, where the numerator denotes the net reward (i.e., the total gain minus the total cost) that the selfish peer i obtained, and the denominator denotes the total number of chunks that i has requested. This utility function represents the average net profit that i can obtain per requested chunk, which i aims to maximize.
   2) For any malicious player $j \in N_m$, its objective is to maximize its utility as in (18), shown at the bottom

of the page. The numerator in (18) represents the net damage caused by j: the first term describes the total costs to other peers when sending the requested chunks to the malicious user j; the middle term evaluates other selfish peers' potential loss in gain due to the incomplete chunk attack by peer j; and the last term is peer j's cost by uploading chunks to other peers. We normalize it using the lifetime of peer j, $T^{(j)}(t_f)$. Now, this utility function represents the average net damage that j causes to the other nodes per time unit.

B. Cheat-Proof and Attack-Resistant Cooperation Stimulation Strategies

Based upon the system description in Section IV-A, we can see that the multiple player game is much more complicated than the two-person game in Section III, and pose new challenges. Thus, direct application of the two-player cooperation strategies to multiple player scenarios may not work.

1) Challenges in Multiple User Scenario: For peer-to-peer live streaming networks in heterogeneous Internet traffic environments, user cooperation stimulation has the following challenges: First, transmission errors are inevitable in the wireless network and the repeated game model is not applicable as discussed int the previous subsection. Second, Malicious users make cooperation stimulation extremely challenging. Misbehavior can result in the decrease of video quality experienced by other peers, which may consequently decrease the quality of service provided by the affected peers. This quality degradation will then be propagated back to the misbehaving peers. Therefore, selfish nodes have no incentives to intentionally behave maliciously in order to enjoy a high quality video. However, the malicious attackers' goal is to degrade the live streaming network performance, and such quality degradation is exactly what they want to see. Unfortunately, malicious behaviors have been heavily overlooked when designing cooperation stimulation strategies.

2) Malicious User Detection: To distinguish "intentional" malicious behavior from "innocent" misbehavior caused by packet delay, we adopt the credit mechanism and the statistical-based malicious user detection in our prior work [20] and introduce trust modelling to resist handwash attack \nocite{Yu:AdHoc}. In this paper, we incorporate the trust modelling into the attacker detection mechanism from the prior work, and will prove by simulation result that the combined anti-attack mechanism can resist handwash attack.

1) **Credit mechanism for pollution attack**: Addressing the pollution attack, for any two peers $i, j \in N$

$$Cc^{(i)}(j,t) = Cu^{(i)}(j,t) - C_p^{(j)}(i,t) \tag{19}$$

calculates the total number of *unpolluted* chunks that user i has uploaded to user j by round t, where $C_p^{(j)}(i,t)$ is the number of polluted chunks that user i has uploaded to user j.

Since peer i cannot identify a chunk as a polluted one until he/she starts decoding and playing that chunk, it is possible that user i *unintentionally* forwards a polluted chunk to other peers. Thus, to distinguish the malicious behavior and the unintentionally pollution by nonmalicious users, we adapt the credit-line mechanism as in our prior work [20] that

$$\begin{aligned} &D^{(i)}(j,t) \le D_{\max}^{(i)}(j,t), \quad \forall t \ge 0, \text{ where} \\ &D^{(i)}(j,t) = Cc^{(i)}(j,t) - Cc^{(j)}(i,t) \\ &\qquad = \left(Cu^{(i)}(j,t) - C_p^{(j)}(i,t)\right) \\ &\qquad\quad - \left(Cu^{(j)}(i,t) - C_p^{(i)}(j,t)\right). \end{aligned} \tag{20}$$

Here, $D_{\max}^{(i)}(j,t)$ is the "credit line" that user i sets for user j at time t. The credit line is set for two purposes: 1) to prevent egoism when favors cannot be simultaneously granted and to stimulate cooperation between i and j, and 2) to limit the possible damages that j can cause to i. By letting $D_{\max}^{(i)}(j,t) \ge 0$, i agrees to send some extra, but at most $D_{\max}^{(i)}(j,t)$ chunks to j without getting instant payback. Meanwhile, unlike acting fully cooperatively, the extra number of chunks that i forwards to j is bounded to limit the possible damages when j plays noncooperatively or maliciously.

To stimulate cooperation in the first few rounds, $D_{\max}^{(i)}(j,t)$ should be large enough in the first few cooperating rounds between user i and j. On the other hand, $D_{\max}^{(i)}(j,t)/[\text{total number of rounds after time } t]$ should be closed to 0 to prevent decreasing the utility of user i. Therefore, when choosing $D_{\max}^{(i)}(j,t)$, user i should first estimate the number of remaining rounds for the live streaming, and choose a relatively small number D_{temp}. Then make D_{temp} with the reciprocal of the probability of successful transmitting a chunk from user j to user i to stimulate the cooperation. A simple solution to this

$$U^{(i)}(t_f) = \frac{\left[Cs^{(i)}(t_f) - \sum_{j\in N} C_p^{(i)}(j,t_f)\right] g_i - \sum_{j\in N} Cu^{(i)}(j,t_f) \frac{MP_i}{B\log(1+P_j A_{ij}^2/d_{ij}\sigma^2)}}{Cr^{(i)}(t_f)} \tag{17}$$

$$U_m^{(j)} = \frac{\sum_{i\in N_s} Cu^{(i)}(j,t_f) \frac{MP_i}{B\log\left(1+\frac{P_j A_{ij}^2}{d_{ij}\sigma^2}\right)} + \sum_{i\in N_s}\left[Cr^{(i)}(j,t_f) - Cs^{(i)}(j,t_f)\right] g_i - \sum_{i\in N} Cu^{(j)}(i,t_f) \frac{MP_j}{B\log\left(1+\frac{P_j A_{ji}^2}{d_{ij}\sigma^2}\right)}}{T^{(j)}(t_f)} \tag{18}$$

is to set the credit lines to be reasonably large positive constants, as in our simulations in Section VI.

2) **Statistical-based malicious user detection**: Since the users have to know the transmission protocol of each other to cooperate, given the signal to noise ratio $P_j A_{ij}/\sqrt{d_{ij}}\sigma^2$, the probability of user j successfully transmit a chunk to user i without retransmission in one round, P_{ij}, can be estimated. P_{ij} can be calculated by the probability successfully transmitting all symbols in a chunk. First, the symbol error rate e_s of each information block given the modulation type, channel coding scheme, and the signal to noise ratio can be analytically calculated according to [22]. Assume there are b_s bits per symbol. Then the P_{ij} can be estimated as $(1-e_s)^{M'/b_s}$. The other way of probing p_{ij} is user i sends probing request to ask user j send the probing package. However, such method is not appropriate in wireless live streaming social network since user j can also intentionally send the incomplete probing package to reduce P_{ij}.

Hence, when player i decides to send a chunk to player j in a round, with probability $1-P_{ij}$, this chunk transmission cannot be completed without retransmission because of the fading channel. That is, we use a Bernoulli random process to model the unsuccessful transmission of a chunk due to high traffic internet connection. Given the Bernoulli random processes and P_{ij} being the probability of successfully receive a chunk in round k, then by time t, user i is supposed to receive $P_{ji} \times Cu^{(j)(i,t)}$ chunks from user j, but the actual number is $C_s^{(i)}(j,t)$. Hence, if user j does not intentionally deploy the incomplete chunk attack, based upon the Lyapunov's Central Limit Theorem [23], if t goes to infinity, then $C_s^{(i)}(j,t) - P_{ij} \times Cu^{(j)(i,t)}$ should follow normal distribution.

For any positive real number x, we can have

$$\lim_{Cu^{(j)}(i,t)\to\infty} \text{Prob}\left(\frac{Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ij}}{\sqrt{Cu^{(j)}(i,t)P_{ij}(1-P_{ij})}} \geq -x\right) = \Phi(x) \quad (21)$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}dt$ is the Gauss tail function.

Therefore, based upon (21), given a predetermined threshold $h > 0$, every selfish peer i can identify peer j as a malicious user by thresholding $Cs^{(i)}(j,t) - Cu^{(j)}(i,t)\overline{P_{ji}(t)}$ as

$$\begin{aligned} &j \in N_m^{(i)}(t), \quad \text{iff} \quad Cs^{(i)}(j,t) \\ &\quad - Cu^{(j)}(i,t)P_{ij} \leq -h\sqrt{Cu^{(j)}(i,t)P_{ij}(1-P_{ij})} \\ \text{and} \quad &j \in N_s^{(i)}(t), \quad \text{iff} \quad Cs^{(i)}(j,t) \\ &\quad - Cu^{(j)}(i,t)P_{ij} > -h\sqrt{Cu^{(j)}(i,t)P_{ij}(1-P_{ij})}. \end{aligned} \quad (22)$$

In (22), $N_m^{(i)}(t)$ is the set of peers that are marked as malicious by peer i at time t, and $N_s^{(i)}(t)$ is the set of peers that are marked as selfish by peer i at time t.

3) **Trust Modelling for handwash attack**: In an environment where malicious users might mount the hand-wash attack, selfish users suffer badly from the hand-wash attack, thus, the unknown risk of interacting with untrustworthy users will reduce the incentive for cooperation in P2P wireless live streaming social networks. With handwash, malicious users can pretend to be innocent until being detected again. The malicious user detection method mentioned previously is statistic-based, which means the selfish users have to wait for enough rounds to interact with the malicious user before detection. This statistics collection process allows the handwashed malicious user to cause extra damage to the system. Thus, to reduce the influence of handwash attack, selfish users have to identify malicious users as soon as possible in order to reduce their losses. A straightforward solution is to reduce the credit line $D_{\max}^{(i)}(j,t)$ defined in (20) or adjust the threshold in (22). However, an arbitrary decrease of the credit line or detection threshold will prevent users from cooperation, resulting in the failure of the whole social network. For instance, if user j is not malicious but just polluted by other malicious users, user i will loose the extra gain by cooperating with user j if user i decreases $D_{\max}^{(i)}(j,t)$ arbitrarily.

Therefore, to provide a guideline of setting the credit line and calculating the detection statistics for malicious users, we introduce the idea of *trust* among selfish users. If a selfish user choose several trusted users to share the information of interaction with other intrusted users, the malicious user detection can be faster, thus, decrease the damage by handwash attack. Also, by taking the damage of the intrusted user j caused to other trusted users into credit line $D_{\max}^{(i)}(j,t)$ can also stop cooperation with malicious users earlier. It is well known that trust is the driving force for cooperation in social networks [24]. In the following we will discuss how to utilize the trust model to against handwash attack. A selfish user i establishes direct trust with another user j upon observations on whether the previous interactions between user i and j are successful. We adopt the beta-function-based method in [25], where user i trusts in user j at time t with value $Tr^{(i)}(j,t)$, which is defined as

$$Tr^{(i)}(j,t) = \frac{Cs^{(i)}(j,t) - C_p^{(i)}(j,t) + 1}{Cr^{(i)}(j,t) + 2}. \quad (23)$$

If user j is not malicious and also not serious polluted, based upon the definition, $Tr^{(i)}(j,t)$ should be closed to P_{ij}. If user j mounts pollution attack, $C_p^{(i)}(j,t)$ will increase and if he/she mounts incomplete-chunk attack, $Cs^{(i)}(j,t)$ will decrease. Thus, both types of attack decrease the numerator in (23), resulting in low trust value for malicious users. Also, the trust is directional, which means user i trusts user j does not imply that user j also trusts user i.

Since the trusted selfish users would like to identify the malicious users together, the damage caused by intrusted users to the trusted users are considered collectively. For example, if user i trusts another user j at round t, user i

consider the damage that malicious user k has caused to user j as his/her own damage. This scenario is equivalent to reduce the credit line $D^{(i)}(k,t)$ in (20) to $D^{(i)}(k,t) - Tr^{(i)}(j,t) \times D^{(j)}(k,t)$. There is an effective bad-mouthing attack against the trust system, where malicious users provide dishonest recommendations to frame up good parties and/or boost trust values of malicious users [24]. To resist such bad-mouthing attack, selfish users should only trust users who have sent them certain number of unpolluted chunks. Assume that selfish user i will only trust user j at time t if user j has sent i more than $Ch^{(i)}(t)$ useful chunks, that is, if $Cs^{(i)}(j,t) > Ch^{(i)}(t)$. The idea for setting $Ch^{(i)}(t)$ is that even the malicious user badmouthes on other selfish users, he/she has to be cooperative and pay enough cost to be trusted, by which the malicious user causes no damage, even contributes, to the system to be trusted. Another advantage of a peer-to-peer cooperation in wireless network is, everyone can listen to the chunk requests and chunk answering of all the users in the network, so the malicious user cannot arbitrarily badmouth the users that he/she has no interaction with.

In summary, the credit line $D^{(i)}_{\max}(j,t)$ in (20) is updated in each round as (24).

If $Cu'^{(i)}(j,t)$ is large enough, the malicious user detection is done at each round by the detection method in (25)

$$D^{(i)}_{\max}(j,t+1) = \max\left\{1, \mathrm{D^{(i)}_{max}(j,t)} - \sum_{k\in N^{(i)}_{\mathrm{Tr}}(t)} Tr^{(i)}(k,t)\times D^{(k)}(j,t)\right\}$$

$$\text{where} \quad N^{(i)}_{\mathrm{Tr}}(t) = \left\{k | k \in N^{(i)}_s(t) \quad \text{and} \quad Cs^{(i)}(k,t) > Ch^{(i)}(t)\right\} \tag{24}$$

$$j \in N^{(i)}_m(t) \text{ iff } Cs'^{(i)}(j,t) - Cu'^{(j)}(i,t)p_{ji} \le -h\sqrt{Cu'^{(j)}(i,t)p_{ji}(1-p_{ji})}, \text{ and}$$

$$j \in N^{(i)}_s(t) \text{ iff } Cs'^{(i)}(j,t) - Cu'^{(j)}(i,t)p_{ji} > -h\sqrt{Cu'^{(j)}(i,t)p_{ji}(1-p_{ji})}, \text{ where}$$

$$Cs'^{(i)}(j,t) = \sum_{k\in N^{(i)}_{\mathrm{Tr}}(t)} Cs^{(k)}(j,t)$$

$$Cu'^{(i)}(j,t) = \sum_{k\in N^{(i)}_{\mathrm{Tr}}(t)} Cu^{(k)}(j,t), \text{ and}$$

$$p_{ji} = \frac{1}{\text{size of } N^{(i)}_{\mathrm{Tr}}(t)} \sum_{k\in N^{(i)}_{\mathrm{Tr}}(t)} \overline{P_{jk}} \tag{25}$$

if $Cu'^{(i)}(j,t)$ is large enough.

As will be demonstrated in Section VI, employing the trust model in (23) and replacing the modified credit line as in (24) will help improve the system's robustness against the handwash attack by malicious users and significantly increase selfish users' utility.

3) Multiuser Cheat-Proof and Attack-Resistant Cooperation Strategy: In summary, the cheat-proof cooperation stimulation strategies in peer-to-peer wireless live streaming social networks are:

Multiuser cheat-proof and attack-resistant cooperation strategy: In the peer-to-peer wireless live streaming game, for any selfish peer $i \in N_s$, he/she initially marks every other user $j \in N, j \neq i$ as selfish. Then, in each round t, i uses the following strategy:

- First bargain the chunk-request quota with other users in the network.
- Update the credit line $D^{(i)}_{\max}(j,t)$ by (24) and identify malicious users by (25).
- If i has been requested by j to send chunks, i will accept this request if j has not been marked as malicious by i and (20) holds; otherwise, i will reject the request.
- When i is requesting a chunk, he/she will send the request to peer j who satisfies

$$j = \arg\max_{\mathrm{j}\in\mathrm{N^{(i)}_s(t)},\mathrm{j}\neq\mathrm{i}} \mathrm{P'_{ji}} \tag{26}$$

where $P'_{ji} = P_{ji} \times Cc^{(i)}(j,t)/Cs^{(i)}(j,t)$ is the probability that user i successfully receives an unpolluted chunk from user j.

C. Strategy Analysis

Using the same analysis as in our prior work [20], the previously mentioned multiuser cheat-proof cooperation strategy can be proven to be a subgame-perfect and Pareto-Optimal Nash equilibrium of the multiuser wireless live streaming game if there exists no attackers. It can also be shown by the proof in [20] that the cooperation strategy is attack-resistant to pollution attack and incomplete chunk attack.

Here we will analyze the optimal attacking strategy with handwash attack.

1) Optimal Attacking Strategy: As discussed in [20], the damage that each attacker by pollution attack and incomplete-chunk attack can cause to selfish user i is bounded by $D^{(i)}_{\max}$, which is negligible if the P2P wireless network has infinite lifetime. In this scenario, peer i will still waste his/her resource on the hand-washed malicious user j since i does not recognize j's new identity and every user is marked as nonmalicious at the beginning. Therefore, with the hand-wash attack, malicious users can increase their payoff dramatically. To simplify the analysis, we assume the attackers will only apply the hand-wash attack at the beginning of each round. For every (selfish or malicious) user in P2P wireless live streaming, at the beginning of each round, besides the strategies discussed in Section IV-A, he/she can also choose to hand wash.

Theorem 1: In the P2P wireless live streaming game where every selfish user follows the cheat-proof cooperation strategy proposed in Section IV-B, if a malicious user i is not detected by any other users and if $D^{(j)}(i,t) < D^{(j)}_{\max}(i,t)$ for all other users $j \in N$, hand wash will not provide the malicious user i any further gain. If the malicious user i is detected by another user j, or if there exists another user $j \in N$ where $D^{(j)}(i,t) \ge D^{(j)}_{\max}(i,t)$, then the hand-wash attack will increase the malicious attacker i's payoff.

Proof: If the malicious user i is not detected by any other user and (20) is satisfied for all $j \in N$, then all the selfish users will still cooperate with the malicious user i. Using the original identity, i receives the same utility as he/she mounts the hand-wash attack and therefore, hand-wash will not bring the malicious user any extra gain. In the scenario where i is detected by a selfish user j as malicious and j refuses to cooperate with i any longer, if i chooses to hand-wash and reenters the game with a new ID, then j will cooperate with i until (20) is not satisfied or i is detected again. Therefore, in this case, i's payoff is increased by causing extra damage to the selfish user j.

From Theorem 1 and [20], *the optimal attacking strategy for a malicious user* is: Upon receiving a request an attacker $j \in Nm$ should always reject the requests; the attackers should always send requests to selfish users, until they do not agree to help, and hand-wash once he/she is identified malicious by one user in the social network. For a malicious use i, to determine whether it has been detected, he/she observes other users' behavior: a selfish user j will always reject the malicious user i's request if and only if i has been identified as malicious by j.

V. P2P Wireless Live Video-Sharing Cooperation Strategy

In this section, we consider two more issues for P2P wireless live video-sharing social networks: coding the live stream into different layers and giving extra chunk-request quota to utilize the broadcast nature of wireless channels. In this paper, we improve the efficiency of cooperation by taking the advantage of the broadcasting nature of the wireless network. Then we present the P2P wireless live video-sharing cooperation strategy.

A. Multiple Layered Coding

Since different users in the P2P wireless live streaming social network use different devices, their demand of video quality is different. For instance, for devices with smaller screen as PDA or cell phones, the spatial resolution of the video can be lower than laptops but still have the same visual quality. Under this circumstances, spatial video coding, which encode the video into bitstreams with different priorities can provide better quality of service. The base layer provide the most important information while the enhancement layers gradually refine the reconstructed video at the decoder's side. Higher layers cannot be decoded without all the lower layers. Therefore, receiving chunks in different layers gives the user different gains, depending upon which video quality the user addresses most.

In addition, suppose that the video is encoded into V_L layers, and based upon user i's device, he/she is satisfied with the video with $V(i)$ layers, then user i has no incentives to ask for chunks in layer higher than $V(i)$. The reason is that since chunks in layer higher than $V(i)$ do not increase visual quality for small-screen device, receiving those chunks gives no gain for user i. Therefore, for each user i, upon deciding which chunks to ask in the round, he/she will first determine how many layers he/she needs based upon the device. Then he/she requests chunks that give him/her the highest video quality depending upon which quality measure user i values most. For the later part of chunk-requesting, we adopt the chunk-request algorithm with tradeoff in our prior work [20].

B. Over-Request for Broadcast Nature

According to the cooperation strategy in Section IV, users will first bargain for chunk-request quota to ensure the total bits to be transmitted in one round does not exceed the channel capacity. On the other hand, the bargained quota also ensures that every user is capable of answering all the requests that he/she receives. Thus, based upon the previously mentioned analysis, selfish users have incentives to answer all the requests in every round.

However, since all users in the peer-to-peer wireless live streaming social network share the same wireless cooperation channel, which has the broadcasting nature that allows the users to listen to others' signals, every selfish user will tend to broadcast the requested chunks to all the users that ask the same chunk to reduce the cost of cooperation. As a result, the overall number of bits transmitted in one round will be much less than the channel capacity since some chunk-requests are combined by one transmission. Therefore, we propose the *over-request* mechanism to fully utilize the channel capacity:

- After bargaining for chunk-request quota, allow each user to send up to K times the bargained quota. $K > 1 \in N$ is is a predefined constant which is agreed by all the users.
- During chunk-requesting stage, users mark the chunk requests with 1 (for the requests use the bargained quota) or 0 (for the requests use the extra quota).
- Then in the request-answering stage, all the users first choose $q = 1$ chunk to be transmitted, and exchange this information to confirm the total bits to be transmitted do not exceed the channel capacity. Increase q until fully utilizing the channel capacity. If when $q = 1$, the total bits to be transmitted exceed the channel capacity, then all the selfish users answer the chunk requests marked with 1.

Although the over-request mechanism can increase the usage of the cooperation channel, the users might not agree to all the chunk requests that are sent to them. Therefore, an algorithm is needed for choosing which chunk requests to answer during cooperation.

Since the live streaming social network will last till the end of the video and has finite life time, selfish users tend to consider the contributions from other peers when choosing which request to answer. This situation will not only encourage the selfish users to be always cooperative in the finite time model but also reduce the damage of handwash attack. Let $Ch^{(i)}(t)$ be the set of chunk indexes that other users request from user i in round t. The users who request chunks from user i must be not marked as malicious by peer i, and also satisfy (20) to make their requested chunks included in $Ch^{(i)}(t)$. We propose the following request-answering algorithm: for every selfish peer i, when he/she receives multiple chunk requests from multiple users and has decided to send q chunks by the previously mentioned over-request mechanism. Then user i chooses q chunks based upon the probability $P^{(i)}(I_j, t)$ defined in (27) where $R(I_k, t)$ is the set of users that request chunk I_k from user i at round t and ϵ is a small number that gives newcomers who have not sent any chunks to peer i a chance to start cooperation.

γ_i is a parameter that controls the sensitivity of user i to other peers' contribution. If $\gamma_i = 0$, every peer sent a request to peer i has the same probability of being answered. On the contrary, if $\gamma_i \to \infty$, the request from user who has send most chunks to peer i will definitely be answered.

C. P2P Wireless Live Video-Sharing Cooperation Strategy With Layered Video Coding and Over-Request

From the previous discussion, the **P2P wireless live video-sharing cooperation strategy** is as follows: for any selfish node $i \in N_s$, he/she initially i marks every other nodes $j \in N$, $j \neq i$ as selfish. Then, in round t, i uses the following strategy:

- Identify malicious users by (22) and update $D^{(i)}_{\max}(i,t)$ by (24).
- Bargain with other users and get the chunk-request quota which is K times the time-sensitive bargaining solution.
- In the chunk-requesting stage, i chooses its own maximum number of video layers $C(i)$ and desired video quality measure, applies the chunk-request algorithm (26), and sends chunk requests to the users in $N_s^{(i)}(t)$.
- Decide q, the number of chunks to transmit in this round by exchanging information with other users in the social network.
- In the request-answering stage, i first identifies the selfish users that satisfy (20). Then, i chooses the chunks to transmit based upon the probability distribution in (27), shown at the bottom of the page, and agrees to send the requested chunks to all the selfish users that ask for the chunks and satisfy (20).

VI. Simulation Results

A. Simulation Settings

We use ns2 and C as the simulation platform. In our simulation, we assume the users communicate with the access point using IEEE 802.11 within the diameter of 15 meters, and users build their own wireless network that uses a different band dedicated to cooperation. ns2 is used to simulate the wired network from the live-streaming server to the access point, and the communication between the access point and the users. The test video is encoded into bit streams and stored sequentially in a file for ns2 to read from. The cooperation among users are simulate by C simulator. ns2 and the C program exchange the real-time simulated results by the log files. The link from the wireless router to the Internet is a DSL link with 1.5 Mbits download bandwidth. There are totally 30 users in the network using live-streaming service, and another 5 users using Internet resources at the same time. For the 5 Internet users, we assume the traffic generated from them is a Poission process. The 30 live-streaming users will cooperate by sharing one channel, and we assume every one in the network can connect with any other user in the network via the dedicated cooperation channel. The location of users are randomly distributed within the circle of 15-meter diameter. The users access the channel by TDMA. We adopt the enter/leave algorithm from the self-organizing TDMA [27]. When a user enters the algorithm, it must first interrupt an existing user's data slot with its firing message. Then the user waits until the beginning of next round to exchange buffer information and join the network, which is, user 1 transmit first then user 2 and so on, and in the next round, user 2 transmit first and user 1 transmit last. After the users have exchanged the requests and decided how many chunks each user is going to transmit, they will transmit in a round sequence based upon the time they join this cooperation. When a user is leaving this network, or in a certain round a user might have nothing to ask from other users, he/she can just keep quiet without doing extra step.

We fix the ratio between the laptop, PDA, and PDA2 users as 3:1:1. The video is initially stored at the original streaming server with an upload bandwidth of 3 Mbps, and there are other 800 users in the Internet watching the same live stream. The request round is 0.4 s and the buffer length is 10 s with $Lf = 20$ and $L = 20$. We choose the "Foreman" and "Akiyo" video sequences with frame rate 30 frames/s. We encode the video using H.264 into a 3-layer bitstream with 75 kbps per layer, and divide each layer into chunks of 0.1 s. Thus, the layered chunk size is $\mathrm{M}' = 7.5$ kbits. In the wireless network, the chunks are channel coded using BCH code with rate 15/31, thus, the chunk size in the wireless live video-sharing social network is $\mathrm{M} = 15.45$ kbits. The 30 live-streaming users in the wireless network can either follow the wireless live streaming cooperation strategy in Section V-C if they are selfish users, and they follow the optimal attack strategy in Section IV-C if they are malicious attackers. We set $g_i = 1 = C_{\max} = 0.8 c_{\mathrm{PDA2}} * K_i$, $c_{\mathrm{PDA2}} : c_{\mathrm{PDA}} : c_{\mathrm{laptop}} = 1 : 0.9 : 0.4$, $P_{\min} = 100$ mW, noise power $= 10$ mW, and bandwidth $\mathrm{B} = 600$ kHz. Discount measure d in (8) is set to be 0.7, γ_i in (27) is set to be 2 and PDA2 and PDA users are satisfied with only receiving the quality of base layer of the video.

The performance of the cooperation strategies is evaluated by the utilities of the users and the PSNR of the video. The PSNR is calculated by first calculating the mean square error between the original video (Foreman or Akiyo) and the received video, and then dividing the peak pixel value by the attained mean square error. If a frame is not received or not decodable, it will introduce the square error equals to the sum of all pixel-value square in the frame.

B. Performance Evaluation

If the attackers mount the hand-wash attack, and the selfish users do not trust each other, the selfish users' utility will be very small no matter which credit line they choose. This case is

$$P^{(i)}(I_j,t) = \sum_{R(I_k,t)\in Ch^{(i)}(t)} \frac{\sum_{m\in R(I_k,t)} \left(Cs^{(i)}(m,t)+\epsilon\right)^{\gamma_i}}{\sum_{R(I_k,t)\in Ch^{(i)}(t)} \sum_{m\in R(I_k,t)} (Cs^{(i)}(m,t)+\epsilon)^{\gamma_i}} \tag{27}$$

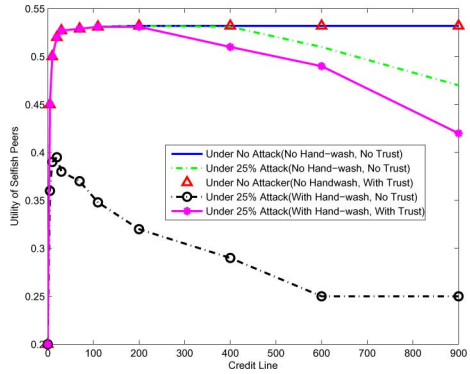


Fig. 6. Utility of selfish (nonmalicious) users under attack versus the initial credit line.

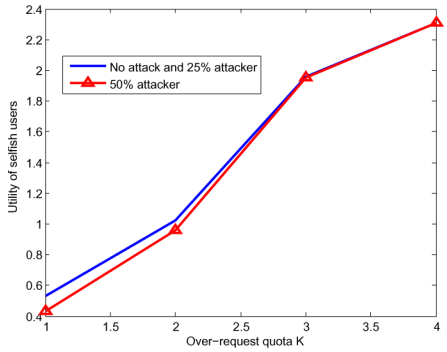
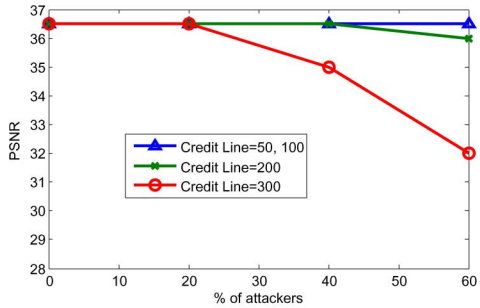


Fig. 7. Utility of averaged selfish (nonmalicious) users with or without attack versus the amount of over-request quota.

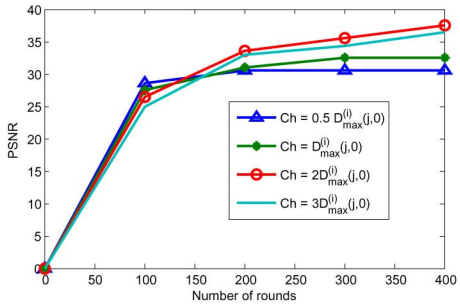
shown as black circle dashed line in Fig. 6. However, the star line in Fig. 6 shows the selfish users' utility if they trust each other, which is much better than without trust. Here we set the minimum number of successfully received chunks $\mathrm{Ch}(i)$ from the trusted users as two times the initial credit line. An intuitive explanation of choosing $\mathrm{Ch}(i)$ is that since the initial setting of credit line $D^{(i)}_{\max}(j,0)$ can be considered as user i's tolerance of the damage that others cause to him/her. On the other hand, $D^{(i)}_{\max}(j,0)$ is the number of chunks that user i thinks an usual nonmalicious user should interact with him/her. Thus, users who have sent more than two times $D(i)_{\max}(j,0)$ chunks successfully to him/her should be trusted. And if the credit line is chosen carefully between 50 and 200, the highest utility can be achieved even the attackers mount the hand-wash attack. The performance of the cooperation strategy with trust when there are no attackers is also presented as red triangle in Fig. 6, showing that trust concept will not degrade selfish users' utility if every one is nonmalicious.

Fig. 7 illustrates the averaged selfish users' utility of the over-request algorithm with or without attack. Here we choose the initial credit line as 50 from the observation drawn in Fig. 6, and set $\mathrm{Ch}(i)$ as 100. When there are 50% of attackers and the users do not over request as in Section V-B, then the utility for selfish users will drop 20% when there are 50% attackers. However, if the users over request to 3 times of the bargained quota, then the utility of the selfish users when there are 50% and 25% attackers will be the same. Thus, it is clear that the over-request algorithm can effectively increase the selfish users utility, and the contribution-based chunk-answering algorithm can also help against attack to 50% malicious users.

Fig. 8 shows the averaged PSNR of the selfish laptop users under different parameter setting. Here the attackers will mount hand-wash attack and the selfish users apply the cooperation strategy as in Section V-C. The PSNR is calculated by the received video given the maximal number of layers of different users. For instance, if the user's device is PDA, then the PSNR is calculated using 2-layer video only. Fig. 8(a) shows the robustness of different credit line setting versus the percentage of attackers. When the percentage of attackers increases, higher credit line setting will give lower PSNR for the selfish users since the credit line mechanism only ensures the maximal damage of each attacker, and the total damage caused by the



(a)

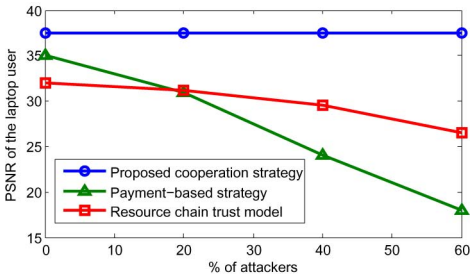


(b)

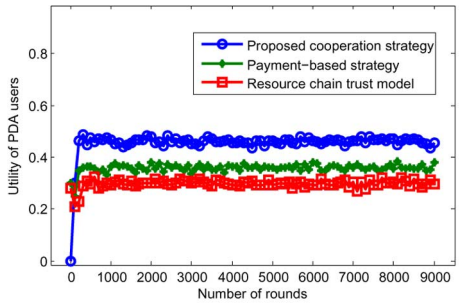
Fig. 8. PSNR of the selfish laptop users: (a) versus percentage of attackers and (b) versus number of rounds.

attackers can increase if there are more malicious users in the system. Thus, this phenomenon again suggests the credit line should be set as the minimal number that can stimulate cooperation, which is 50 in this case. Fig. 8(b) shows the selfish user's averaged PSNR under different trust thresholds Ch in (24) versus the number of rounds. It is clear after 400 rounds that the selfish user's PSNR is saturated and $\mathrm{Ch} = 0.5D^{(i)}_{\max}(j,0)$ or $\mathrm{Ch} = D^{(i)}_{\max}(j,0)$ gives lower PSNR than $\mathrm{Ch} = 2D^{(i)}_{\max}(j,0)$. These results imply that setting trust threshold Ch too small will cause damage to the system since the selfish users might trust the malicious users also. On the other hand, from Fig. 8(b), higher Ch needs more number of rounds to saturate the selfish user's PSNR, which means the selfish users need to wait more rounds to trust other users.

Furthermore, we compare our cooperation strategy with the payment-based incentive schemes [13] and the resource chain trust model for P2P security [28]. The credit line is set to 100,



(a)



(b)

Fig. 9. Performance comparison of the proposed cheat-proof and attack-resistant cooperation strategies and the payment-based cooperation strategy and the resource chain trust model: (a) PSNR of laptop users versus percentage of attackers and (b) utility of PDA users versus number of rounds.

and the users over request the chunk by 3 times. We first compare the attack-resistance of the three algorithm as shown in Fig. 9(a). It is clear that our cooperation strategy is attack-resistant when the percentage of attackers is less than 60%, and the resource chain trust model can resist up to 30% of attackers. The payment-based method is not resistant to the attack, while under no attack the payment-based method can achieve 35 dB but still lower than the proposed cooperation strategy since the payment-based method does not consider the issues of wireless channels.

We also compare the utility for the PDA user versus number of rounds for the three algorithms without attack in Fig. 9(b). First, the proposed algorithm converge to steady payoff as quick as the payment-based method, while the resource chain trust modelling takes longer time. On the other hand, our proposed scheme gives the PDA users higher utility by taking into account the desired resolution of the user. The PDA user will not request higher-layer chunks and, thus, he/she will dedicate his/her chunk-request quota to the base-layer chunks and get higher utility.

VII. Conclusion

In this paper, we investigate cooperation stimulation in wireless live-streaming social networks under a game theoretic framework. An illustrating two-player Bayesian game is studied, and different optimality criteria, including Pareto-Optimal and time-sensitive bargaining solution is performed to refine the obtained equilibriums. Finally, a cheat-proof cooperation strategy is derived which provides the users in wireless live streaming social network an secured inventive to cooperate. The results are then extended to stimulate multiuser live streaming, and combing with the chunk-request and request-answering algorithm, a fully-distributed attack-resistant and cheat-proof cooperation stimulation strategy has been devised for peer-to-peer wireless live streaming social networks. Simulation results have illustrated that the proposed strategies can effectively stimulate cooperation among selfish peers in a wireless network, and the incentive-based cooperation strategies are attack-resistant to pollution attack and handwash attack when the percentage of attackers is less than 25%.

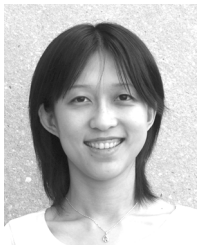
References

[1] [Online]. Available: http://www.pplive.com/en/index.html
[2] S. C. Kim, M. G. Kim, and B. H. Rhee, "Seamless connection for mobile p2p and conventional wireless network," in *Proc. Int. Conf. Advanced Communication Technology*, Feb. 2007, vol. 3, pp. 1602–1605.
[3] M. Cagalj, S. Capkun, and J.-P. Hubaux, "Key agreement in peer-to-peer wireless networks," *Proc. IEEE*, vol. 94, no. 2, pp. 467–478, Feb. 2006.
[4] S. Ghandeharizadeh, B. Krishnamachari, and S. Song, "Placement of continuous media in wireless peer-to-peer networks," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 335–342, Apr. 2004.
[5] S. Ghandeharizadeh and T. Helmi, "An evaluation of alternative continuous media replication techniques in wireless peer-to-peer networks," in *Proc. 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access*, New York, 2003, pp. 77–84.
[6] G. Owen, *Game Theory*, 3rd ed. New York: Academic, 2007.
[7] O. Karonen and J. K. Nurminen, "Cooperation incentives and enablers for wireless peers in heterogeneous networks," in *Proc. IEEE Int. Conf. Communications*, Beijing, China, May 2008, pp. 134–138.
[8] T. Ozbilgin and M. O. Sunay, "On the capacity of wireless peer-to-peer networks with transmitter and receiver cooperation," in *Proc. IEEE Int. Symp. on Personal, Indoor and Mobile Radio Communications*, Athens, Greece, Sep. 2007, pp. 1–5.
[9] C. Buragohain, D. Agrawal, and S. Suri, "A game theoretic framework for incentives in p2p systems," in *Proc. Int. Conf. Peer-to-Peer Computing*, Sep. 2003, pp. 48–56.
[10] S. Jun and M. Ahamad, "Incentives in bittorrent induce free riding," in *Proc. 2005 ACM SIGCOMM Workshop on Economics of Peer-to-Peer Systems*, 2005, pp. 116–121.
[11] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *Proc. SIGCOMM 2004*, 2004, pp. 367–378.
[12] A. Habib and J. Chuang, "Incentive mechanism for peer-to-peer media streaming," in *Proc. Int. Workshop on Quality of Service (IWQoS)*, Jun. 2004, pp. 171–180.
[13] G. Tan and S. A. Jarvis, "A payment-based incentive and service differentiation mechanism for peer-to-peer streaming broadcast," in *Proc. International Workshop on Quality of Service (IWQoS)*, Jun. 2006, pp. 41–50.
[14] Z. Liu, Y. Shen, S. Panwar, K. Ross, and Y. Wang, "Using layered video to provide incentives in p2p live streaming," ACM Special Interest Group on Data Communication Aug. 2007.
[15] J. Liang, R. Kumar, Y. Xi, and K. Ross, "Pollution in P2P file sharing systems," in *Proc. IEEE Conference on Computer Communications*, Dec. 2005, vol. 2, pp. 1174–1185.
[16] N. Naoumov and K. Ross, "Exploiting P2P systems for DDoS attacks," in *Proc. 1st Int. Conf. Scalable Information Systems*, 2006, p. 47.
[17] G. Kramer, M. Gastpar, and P. Gupta, "Cooperative strategies and capacity theorems for relay networks," *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3037–3063, Sep. 2005.
[18] X. Hei, Y. Liu, and K. W. Ross, "Inferring network-wide quality in {P2P} live streaming systems," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 9, pp. 1640–1654, Dec. 2003.
[19] M. J. Osborne and A. Rubinste, *A Course in Game Theory*. Cambridge, MA: MIT Press, 1994.
[20] W. S. Lin, H. V. Zhao, and K. J. R. Liu, "Incentive cooperation strategies for peer-to-peer live streaming social networks," *IEEE Trans. Multimedia*, vol. 11, no. 3, pp. 396–412, Apr. 2009.
[21] W. Yu and K. J. R. Liu, "Game theoretic analysis of cooperation and security in autonomous ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 5, pp. 507–521, May 2007.

[22] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Upper Saddle River, NJ: Prentice-Hall, 1983.
[23] O. Kallenberg, *Foundations of Modern Probability*. New York: Springer-Verlag, 1977.
[24] Y. Sun, Z. Han, and K. J. R. Liu, "Defense of trust management vulnerabilities in distributed networks," *IEEE Commun. Mag.*, vol. 42, no. 2, pp. 112–119, Feb. 2008.
[25] A. Josang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Syst.*, vol. 42, no. 2, pp. 618–644, 2005.
[26] W. S. Lin, H. V. Zhao, and K. J. R. Liu, "Cheat-proof cooperation strategies for wireless live streaming social networks," in *Proc. IEEE Int Conf. Acoustic, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, R.O.C., 2009, pp. 3469–3472.
[27] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "Desync: Self-organizing desynchronization and TDMA on wireless sensor networks," in *Proc. 6th Int. Conf. Information Processing in Sensor Networks*, 2007, p. 20.
[28] S. Lee, S. Zhu, and Y. Kim, *P2p Trust Model: The Resource Chain Model*, vol. 2, pp. 357–362, 30 2007-Aug. 1 2007.

W. Sabrina Lin (M'06) received the B.S. and M.S. degrees from National Taiwan University, Taipei City, Taiwan, R.O.C., in 2002 and 2004, respectively, and the Ph.D. degree from University of Maryland, College Park, in 2009, all in electrical engineering.

Her research interests are in the area of information security and forensics, multimedia signal processing and multimedia social network analysis. She received the University of Maryland Future Faculty Fellowship in 2007.



H. Vicky Zhao (M'05) received the B.S. and M.S. degree from Tsinghua University, Beijing, China, in 1997 and 1999, respectively, and the Ph.D. degree from University of Maryland, College Park, in 2004, all in electrical engineering.

She was a Research Associate with the Department of Electrical and Computer Engineering and the Institute for Systems Research, University of Maryland, College Park from January 2005 to July 2006. Since August 2006, she has been an Assistant Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Canada. Her research interests include information security and forensics, multimedia social networks, digital communications and signal processing.

Dr. Zhao received the IEEE Signal Processing Society (SPS) 2008 Young Author Best Paper Award. She co-authored the book "Multimedia Fingerprinting Forensics for Traitor Tracing" (Hindawi, 2005). She is an Associate Editor for IEEE SIGNAL PROCESSING LETTERS and *Journal of Visual Communication and Image Representation*.

K. J. Ray Liu (F'03) is a Distinguished Scholar-Teacher of University of Maryland, College Park. He is Associate Chair of Graduate Studies and Research of Electrical and Computer Engineering Department and leads the Maryland Signals and Information Group conducting research encompassing broad aspects of wireless communications and networking, information forensics and security, multimedia signal processing, and biomedical engineering. His recent books include *Cognitive Radio Networking and Security: A Game Theoretical View* (Cambridge University Press, 2010); *Cooperative Communications and Networking* (Cambridge University Press, 2008); *Resource Allocation for Wireless Networks: Basics, Techniques, and Applications* (Cambridge University Press, 2008); *Ultra-Wideband Communication Systems: The Multiband OFDM Approach* (IEEE-Wiley, 2007); *Network-Aware Security for Group Communications* (Springer, 2007); *Multimedia Fingerprinting Forensics for Traitor Tracing* (Hindawi, 2005); and *Handbook on Array Processing and Sensor Networks* (IEEE-Wiley, 2009).

Dr. Liu is President-Elect and was Vice President—Publications of IEEE Signal Processing Society. He was the Editor-in-Chief of IEEE SIGNAL PROCESSING MAGAZINE and the founding Editor-in-Chief of *EURASIP Journal on Advances in Signal Processing*. He is the recipient of numerous honors and awards including IEEE Signal Processing Society Technical Achievement Award and Distinguished Lecturer. He also received various teaching and research recognitions from University of Maryland including university-level Invention of the Year Award; and Poole and Kent Senior Faculty Teaching Award and Outstanding Faculty Research Award, both from A. James Clark School of Engineering. Dr. Liu is a Fellow of IEEE and AAAS.