Discouraging Selfishness in Lossy Peer-to-Peer Networks

Alex Friedman and Idit Keidar*

January 2009

Abstract

We present *Loss-Tolerant Selfishness Monitor (LTSM)*, a generic service for detecting selfish behavior in various P2P applications, such as MANET routing and multicast. Unlike most previous selfishness-resistant protocols, LTSM can be used in networks subject to message loss, where selfish behavior detection is particularly challenging. One of our main contributions is mathematically analyzing the impact of various system parameters on the incentives for cooperation, and showing how to choose these parameters so as to ensure full cooperation at a minimal cost. We illustrate the applicability of LTSM in two exemplar contexts: multicast and MANET routing.

1 Introduction

Peer-to-Peer (P2P) protocols are used in numerous different settings, e.g., mobile-ad-hoc networks (MANETs), P2P multicast systems, and file sharing networks. The underlying networks used by many such P2P systems are lossy. For example, wireless networks, such as MANETs, inherently suffer from high packet loss rates. Furthermore, multicast systems for streaming video or audio typically use unreliable transport like UDP, since it is acceptable for some of the data to be lost.

Resources in P2P systems are provided by the participating peer nodes themselves; each node has to contribute memory, CPU power, bandwidth, and energy. Since most nodes in a MANET are battery-powered, energy is a scarce resource in such an environment. In commercial P2P applications, nodes may exhibit selfish behavior by tampering with the P2P protocol in order to lower their cost [3, 6, 18]. Consequently, it is important for such protocols to work well even when users are equipped with a selfish version of the protocol.

In recent years, much research has been dedicated to tackling selfish behavior in various P2P applications (e.g., MANET routing, multicast, and file sharing – see Section 2). Many challenging issues, however, remain open. Previous work, for instance, has not exposed and leveraged the similarity among different P2P protocols. Rather, each previous work has focused on one specific protocol, in one specific setting. Another challenge largely overlooked in previous work is lossy networks (with the exception of [26, 27] – see Section 2). Selfish behavior detection becomes much more challenging when one has to cope with unpredictable packet loss. Conventional detectors, such as those used in [2, 4, 18, 19], would wrongfully accuse cooperating nodes for not sending lost packets. Finally, previous work has not mathematically quantified the relationship that needs to hold among system parameters such as a cooperating node's cost, the penalty for lack of cooperation, and the decision when to punish a node, in order to achieve full cooperation at a minimal cost.

In this paper, we address the three open issues above. We leverage the similarity among many different P2P protocols in order to define a common monitoring abstraction suitable for detecting selfish behavior in various such protocols (Section 3). Our abstraction's interface enables each peer node to monitor other nodes, and to determine when to punish a node for alleged misbehavior. We then present a

*Department of Electrical Engineering, Technion, Haifa, 32000, Israel. {ghost@tx, idish@ee}.technion.ac.il

loss-tolerant selfishness monitor (LTSM) that implements this interface (Section 4). When using LTSM, a node blamed for lack of cooperation must pay a fine to continue participating in the protocol.

One of the main contributions of this paper is mathematically quantifying the relations between the above fine, the cost of performing a basic operation (such as sending a message), the packet loss rate, the decision as to when to punish a node, and the costs incurred by cooperative and non-cooperative nodes (Section 5 and Appendix A). We show how to tune LTSM's parameters so as to make full cooperation and fully following the protocol a Nash equilibrium, while minimizing the cost for cooperating nodes.

To illustrate the applicability of our abstraction, we show (Section 6) how it can be seamlessly employed in existing multicast schemes [5, 14, 21], and in existing schemes for MANET routing [2, 4, 11, 18, 19]. By using LTSM with the appropriate parameter settings in these applications, one can automatically make them robust to packet loss.

2 Related Work

Selfish behavior has been widely studied in various P2P systems, e.g., content distribution protocols [6, 10], tree-based multicast [21], gossip-based multicast [14, 17], and MANET routing [2, 4, 7, 11, 18, 19]. A game-theoretic approach has been used in previous work to provably enforce cooperation [1, 9, 14, 16, 17, 20, 24, 26, 27]. Each existing solution, however, is built for a specific application. Furthermore, each current MANET routing solution deals with one specific routing protocol, usually source routing (e.g., DSR [13]). In contrast, the solution we present here is general, and suitable for a wide range of P2P applications.

By and large, previous work has not taken message loss into account. The only exception we are familiar with is the MANET routing scheme due to Yu and Liu [26, 27]. There are several important differences between their work and ours. Whereas Yu and Liu focus on stimulating cooperation in a MANET source routing (DSR) protocol, we provide a general abstraction for P2P systems. Furthermore, Yu and Liu focus solely on packet forwarding, while LTSM allows for monitoring additional message types, such as route discovery, location queries and replies, and keep-alive messages. Third, their work employs a tit-for-tat (TFT) strategy between each pair of nodes in the network. In this strategy, a node agrees to forward packets on behalf of another node only if the latter has previously forwarded enough packets for the former. We, on the other hand, do not assume any specific strategy. Fourth, their analysis only shows how to set the cooperation threshold for a given desired false positive probability. They do not analyze how the false positive probability should be chosen so as to ensure cooperation at a minimal cost for cooperative nodes as we do. Moreover, their punishment scheme is quite draconic as there is no way for selfish nodes to be added back, which suggests that the false positive probability should be chosen to be very small. In contrast, we allow nodes suspected of non-cooperation to be added back by paying a fine.

3 Monitoring Service Definition

We consider a P2P system in which the participating nodes are selfish and rational, i.e., each node wishes to participate in the protocol while choosing a strategy that minimizes its cost. A strategy consists of deciding which packets to transmit, out of the packets required by the P2P protocol. We say that a node is *cooperative* if it sends all the packets required by the protocol, and *non-cooperative* otherwise. For simplicity, we assume that the cost of sending all packets is the same.

Nodes may join and leave the system dynamically. Nodes can be removed from the system, e.g., due to misbehavior, and may be allowed back after paying an application-defined *fine*, specified in terms of the packet sending cost. For example, $fine = 7$ means that a suspected node has to send seven penalty packets in order to be allowed back into the system. As free admission can be abused in systems subject

to Sybil attacks [8], some sort of payment is required in order to join such a system. One can set the cost of joining to be the same as the fine.

We classify messages in a P2P protocol into two categories. The first category consists of messages that are generated by nodes at a predetermined rate, one per a given *time unit*. Examples include keep-alive messages and data packets in a Constant Bitrate (CBR) stream. A CBR stream is possible in a lossy environment if the protocol requires each node to compensate for lost packets by sending empty packets instead, as done, for example, in [14, 17]. The second category includes request-based messages, whose sending is triggered by the receipt of other messages at unpredictable times, e.g., forwarding data in a routing protocol or sending a piece of content in response to a request in a content distribution protocol.

We consider a lossy underlying network in which packet loss is independent and identically distributed (i.i.d.) with probability p. Note that whenever a node detects the loss of its own packet, it may simply retransmit the packet to avoid being suspected of selfish behavior. We therefore restrict our attention to the case that a node cannot detect whether its own message has been lost.

Our goal is to provide a monitoring service for P2P applications that guarantees the cooperation of rational nodes while minimizing the expected cost of cooperative nodes, and taking message loss into account.

The interface of our monitoring service is as follows: *start_monitor(N)* is invoked by the application when either a new node, N, is discovered, or when an allegedly non-cooperative node rejoins the system after paying a fine; *missed_message(N)* is called when a message the P2P application is expecting from N does not arrive in a timely fashion, and *detected_message(N)* is called when such an expected message is detected on time. Lastly, the *is_selfish(N)* predicate indicates whether node N is allegedly non-cooperative.

4 LTSM Algorithm

The LTSM service running at every node keeps an activity record for every monitored node. A node is deemed non-cooperative if it has sent less than τ out of W expected messages, where W is a parameter window size, and τ is the detection threshold. The windows in the protocol do not overlap – all counters are reset after each window of size W. W and τ are measured in number of messages. The relations between W, τ, and the fine are analyzed in Section 5.

Algorithm 1 LTSM(W, τ)

start_monitor(N)

1: $X[N] \leftarrow 0$ {expected packets}
2: $Y[N] \leftarrow 0$ {detected packets}
3: *is_selfish[N]* $\leftarrow$ **false**

is_selfish(N)

1: **return** *is_selfish[N]*

detected_message(N)

1: **if** *is_selfish[N]*=**false then**
2: $Y[N] \leftarrow Y[N] + 1$
3: *advance_window(N)*

missed_message(N)

1: **if** *is_selfish[N]*=**false then**
2: *advance_window(N)*

advance_window(N)

1: $X[N] \leftarrow X[N] + 1$
2: **if** $X[N]$=W **then**
3: **if** $Y[N] \leq \tau$ **then**
4: *is_selfish[N]* $\leftarrow$ **true**
5: **else** {start a new window}
6: $X[N] \leftarrow 0$
7: $Y[N] \leftarrow 0$

The LTSM protocol is depicted in Algorithm 1. The activity record of each monitored node, N, consists of two counters and a boolean, *is_selfish[N]*, which indicates whether the node is allegedly non-cooperative. The first counter, $X[N]$, counts the number of packets expected in the current window. The second counter, $Y[N]$, counts the number of packets detected in the current window. The

start_monitor(N) method resets all these counters to zero and sets *is_selfish[N]* to *false*. The *is_selfish(N)* method simply returns the boolean *is_selfish[N]*. Recall that the *missed_message(N)* method is called by the P2P application when an expected message does not arrive on time. Hence, this method advances the window by increasing the expected messages counter, $X[N]$. A decision is made whether node N should be deemed non-cooperative at the end of the window, i.e., when the $X[N]$ counter reaches W. If N is declared cooperative (not selfish), then all counters are reset, and a new window begins. Lastly, recall that the *detect_message(N)* method is called when an expected message is detected. This method increases the detected messages counter, $Y[N]$, and then advances the window as in *missed_message(N)*. Notice that no counters advance in case the node is alleged non-cooperative.

Algorithm 2 shows two generic usage examples for LTSM, one of a CBR stream, and one of request-based traffic. We leave out the initializations, and simply assume that *start_monitor(N)* has been called for each node N. In a CBR stream, a timer is used to check whether a packet arrives at every (predetermined) time unit; *detected_message(N)* is called if such a packet arrives, and *missed_message(N)* is called otherwise. Similarly, for request-based traffic, a timeout is used to determine whether a given request yields a response. An allegedly non-cooperative node is denied service until it pays a fine. It is important to note that LTSM's parameters (W, *fine*, and τ) differ for CBR and request-based traffic monitoring (see Section 5). More detailed usage examples are provided in Section 6.

Algorithm 2 LTSM generic usage examples

on request to serve(N)
1: **if** *is_selfish[N]*=**false then**
2: serve request

CBR(N)
every time unit do
1: **if** received packet **then**
2: *detected_message(N)*
3: **else**
4: *missed_message(N)*

on receive fine from N
1: *start_monitor(N)*

Request-based(N)
on detect request do
1: wait timeout
2: **if** received response packet **then**
3: *detected_message(N)*
4: **else**
5: *missed_message(N)*

5 Analysis

Our goal is to understand the relations between W, τ, the fine, the loss rate p, and the expected costs of cooperative and non-cooperative nodes. These relations help us determine the best parameter choices for the protocol, so that full cooperation and fully following the protocol is a Nash equilibrium. We define $q = 1 - p$. We first analyze constant-rate traffic (CBR) in Section 5.1, then provide several graphs with numerical examples in Section 5.2, and finally extend the analysis to request-based traffic (RB) in Section 5.3. For readability of this section, formal proofs are deferred to Appendix A.

5.1 Analysis for Constant Rate Traffic

Due to the CBR nature of the traffic, a single packet is expected to arrive per time unit. As packet loss cannot be detected by the sending node itself, the decision whether to send a given packet is independent of previous events. We therefore assume that each node decides in advance on the number of packets it sends in a given window.

Consider a node I monitoring another node F. We use the following notations:

$$y(X) \triangleq \text{the number of packets received at I out of X packets sent by F}$$
$$D(X,\tau) \triangleq P(y(X) \leq \tau), \text{ detection probability}$$
$$\epsilon(\tau) \triangleq D(W,\tau) = P(y(W) \leq \tau), \text{ false positive probability}$$

Let $y(X)$ be a random variable representing the number of packets that are received at I in a given window, out of X packets sent by F. Since loss is i.i.d. with probability $p = 1 - q$, $y(X)$ is a binomial random variable, *y(X)∼Binomial(X,q)* [25]. F is detected as faulty unless more than τ packets arrive. The detection probability is therefore equal to the binomial cumulative distribution function at τ,

$$D(X,\tau) = P(y(X) \leq \tau) = \begin{cases} I_p(X-\tau, \tau+1) & \text{if } X > \tau, \\ 1 & \text{otherwise,} \end{cases} \tag{1}$$

where I is the *regularized incomplete beta function* [25].

We now turn to compute the expected cost of sending X packets in a window. Recall that the sender has to pay a fine to continue participating, in case the number of packets received, $y(X)$, is lower than the detection threshold, τ. Thus, the expected cost, $Ecost(X,\tau)$, when sending X packets is:

$$Ecost(X,\tau) = X + fine \times D(X,\tau). \tag{2}$$

Our goal is to find *fine*, *W*, and τ that encourage full cooperation, i.e., ensuring that for a given *W*, and all natural $1 \leq n \leq W$, $Ecost(W-n,\tau) > Ecost(W,\tau)$. That is, the expected cost of sending less than W messages is strictly higher than the expected cost of sending W messages, assuming that the sender intends to continue participating in the protocol. The required relations among the parameters are captured by the following lemma, which is proved in Appendix A.1.

Lemma 1. *For all natural n, $0 < n \leq W$: $Ecost(W-n,\tau) > Ecost(W,\tau)$ if and only if the following constraint holds:*

$$fine > \max\left(\frac{W}{1-\epsilon(\tau)}, \frac{1}{qP(y(W-1)=\tau)}\right).$$

Lemma 1 provides a constraint that must be upheld in order to enforce full cooperation. We now turn to minimizing the expected cost of a cooperating node under this constraint. This is a discrete optimization problem.

To solve this problem, we convert it to a continuous problem by using an approximation for large values of W. A common rule-of-thumb is that if both Wp and Wq are greater than 5 [15], then the binomial distribution can be approximated by the normal distribution. E.g., for a loss probability of $p = 0.1$, $W = 50$ should suffice.

Somewhat surprisingly, the following lemma (Lemma 2) shows that the expected cost for a cooperative node is minimized at $\tau = 0$, i.e., a single packet arriving at the destination is sufficient for the node to be considered cooperative. This, however, is obtained at the expense of a very high $fine$ value. The following lemma is proved in Appendix A.2.

Lemma 2. *For a large enough W, under the constraint of Lemma 1, the expected cost for a cooperative node is minimized at $\tau = 0$. The minimal expected cost is $W + p/q$, and the fine has to satisfy $fine > 1/(qp^{W-1})$.*

Although Lemma 2 is stated (and proven) only for large values of *W*, we have empirically observed that the result actually holds for all values of W.

Though Lemma 2 identifies the optimal fine in terms of overall cost, it calls for very high fines. For example, with $p = 0.1$ and $W = 50$, the fine, according to Lemma 2, should be higher than 10^{49}. In

Section 5.2, we illustrate some numerical examples, and show that W has a similar influence on the cost: the higher the fine and the window size are, the lower the expected cost for cooperative nodes is. On the other hand, with a high W, it takes LTSM a long time to detect a non-cooperative node. Similarly, a high fine can be detrimental for performance, especially in systems that are susceptible to Sybil attacks, where a high fine would entail a high join cost. Thus, there is a trade-off involved in setting these parameters.

A typical P2P application would therefore optimize its cost under additional constraints on the highest acceptable fine and W. Nevertheless, Lemma 1 requires the fine to be greater than $W/(1-\epsilon(\tau))$, which is at least greater than W. Thus, not all values of $fine$ and W are feasible.

Let erf be the error function [25], and

$$\tau_{min} \triangleq qW - 0.5 - \sqrt{2pqW}\sqrt{-\ln\left(\frac{\sqrt{2\pi(W-1)pq}}{q \times fine}\right)}, \tag{3}$$

$$\delta_{min} \triangleq \frac{1 - \mathrm{erf}\left(\sqrt{-\ln\left(\frac{\sqrt{2\pi(W-1)pq}}{q\times fine}\right)}\right)}{2}. \tag{4}$$

The following lemma shows that setting the threshold value (τ) to τ_{min} defined above warrants full cooperation and yields a minimal expected cost, given pre-defined W and $fine$, and assuming that the constraint of Lemma 1 holds, which is captured using δ_{min} above.

Lemma 3. *Given $fine$ and W such that W is large, if $fine > W/(1-\delta_{min})$, and $\delta_{min} < 0.5$, then choosing $\tau = \lfloor \tau_{min} + 1 \rfloor$ warrants full cooperation and yields a minimal expected cost over all possible values of τ.*

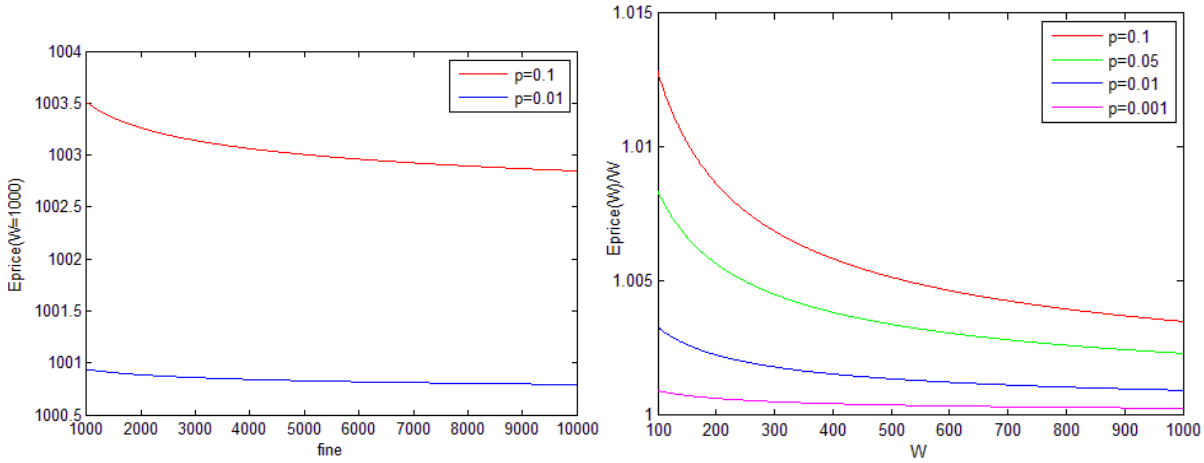
We prove the above lemma in Appendix A.3. Notice that feasible values of W and $fine$ can be found numerically using Lemma 3.

5.2 Numerical Examples

For fixed $fine$ and W, we compute the optimal τ using Lemma 3. Figure 1a depicts the resulting expected cost for $W = 1000$ as a function of the fine for two loss probabilities ($p = 0.1$, $p = 0.01$). We see that the higher the fine is, the lower the expected cost is (as predicted by Lemma 2). However, choosing a fine significantly lower than the optimal according to Lemma 2 is not too costly, as the expected cost decreases rather slowly as $fine$ increases. The reasoning behind this behavior is that for a higher $fine$, a lower τ is sufficient to discourage selfish behavior. A lower τ results in a lower false positive probability ,$\epsilon(\tau)$, which decreases super-linearly in $fine$.

Figure 1b depicts the normalized expected cost, i.e., $Ecost(W,\tau)$ divided by W, for various window sizes, using the smallest fine satisfying the constraint of Lemma 1. We see that increasing W significantly reduces the normalized expected cost. Notice that the value for δ_{min} as defined in (4) grows sub-linearly in W. Recall that the expected cost for a cooperative node is equal to $W + \delta \times fine$. Thus, if only $fine$ is constrained by the application (and not W), then the largest W that is feasible according to Lemma 3 minimizes the normalized expected cost, $Ecost(W,\tau)/W$.

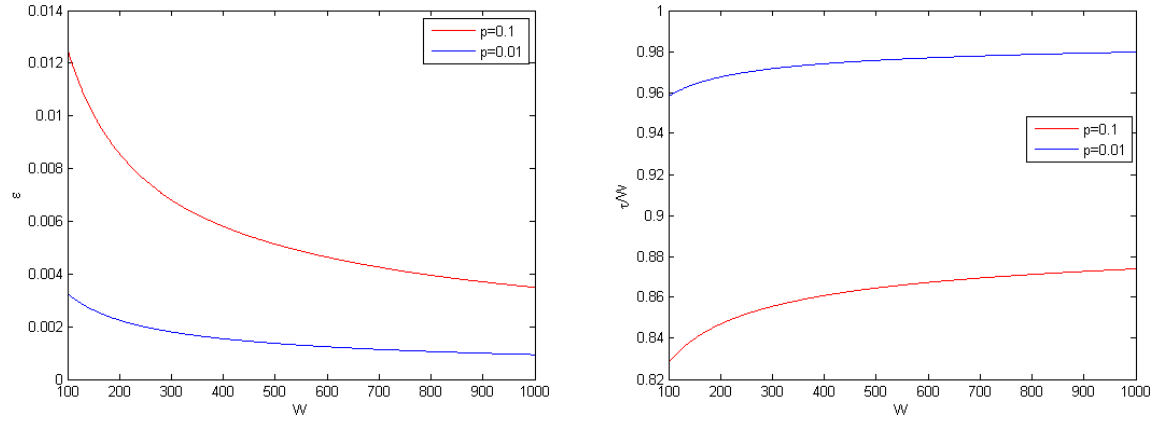
The false positive probability decreases rapidly as W increases, as we can see in Figure 2a. We use the smallest fine satisfying the constraint of Lemma 1 for each W. The reasoning behind this behavior is that τ/W increases in W, as can be seen from (3), which lowers the false positive probability. Figure 2b shows that the detection threshold, τ, becomes closer and closer to W as W increases, when using the smallest fine as described above.



(a) Expected cost for W=1000

(b) Expected cost divided by W for the minimal possible fine

Figure 1: The impact of W, fine, and p on the expected cost.



(a) False positive probability

(b) Detection threshold, τ, divided by W

Figure 2: The effect of W.

5.3 Analysis for Request-Based Traffic

We now adapt our previous discussion to request-based traffic (such as data forwarding in a routing protocol). Let I be an inspecting node at which we measure the counters, X and Y, and let F be the inspected node. A request message that should trigger request-based sending may be issued by I itself, or, in some applications, by some other node S. In any case, I must be able to detect both the request and the response messages. Upon detecting a request message, node I may falsely accuse F of a packet loss in two cases:

1. F does not receive the request message; or
2. F receives the request message, but I does not receive F's response.

Hence, the probability that I falsely accuses F is $\tilde{p} = p + qp = 1 - q^2$, instead of p as in the previous section.

Let W be the window size (in messages). As in the previous section, I decides whether a node is behaving selfishly or not at the end of each such window. The threshold, τ, and the fine value that ensure full cooperation are obtained the same way as in the previous section, by using $\tilde{p}$ instead of p.

Note that in some applications there may be messages that F is asked to send by S which are unseen by I. Such messages may safely be dropped by F without fearing detection by I. However, in most cases, F has no way of knowing whether I has heard a message or not. In some cases, such as a MANET in which node's locations and radio ranges are known, F may drop packets knowing it won't be detected by I. On the other hand, it may be detected by other nodes.

6 Applications

In this section we show two examples of how LTSM can help make P2P algorithms immune to rational selfish nodes, namely, multicast (Section 6.1) and MANET routing (Section 6.2).

6.1 Multicast

We first focus on tree-based multicast (either wired or wireless), which is the most common multicast architecture. We follow the common practice of assuming that the multicast source node is altruistic and trusted by all nodes [14, 17, 23]. The multicast protocol disseminates messages over an overlay tree, which spans all participating nodes. One way to build and maintain the multicast tree is using a trusted entity, such as the multicast source [12, 22], which eliminates selfish behavior in the tree-building stage. A distributed tree building scheme which is immune to selfish behavior (such as in [21]) can also be used. The tree must be changed dynamically to account for topology changes, which are common, especially in MANETs. For the remainder of this section, we assume an external mechanism for building the multicast tree.

Algorithm 3 employs the interface defined in Section 3 to monitor and punish non-cooperative nodes. Let r be the multicast rate. For simplicity, we assume that the multicast application must receive exactly one packet every $1/r$ seconds and that the packets are numbered. A node receives the current packet number and its parent's name from the source when it joins a multicast tree. It then uses LTSM to monitor the traffic forwarded by its parent. A message miss is registered if no packet is received from the parent in a given interval. A message detection is registered otherwise. A node that misses a message that it has to send to its children, compensates for it by sending a dummy packet instead. We assume that the cost of sending dummy packets is the same as that of sending data packets, and hence a selfish rational node has no reason to send dummy packets when it does have data to send.

Algorithm 3 Monitoring Service for Tree-Based Multicast

Source

on_suspicion_report(P)

1: **if** reporter is not child of P **then**
2: **return** {ignore}
3: ask P to pay a fine
4: **if** not received a fine **then**
5: Ask *parent(P)* to Disconnect P
6: Reconnect P's sub-tree

Client Node

on_join_tree(*P*, *cur_packet*)

1: *start_monitor(P)*
2: $i \leftarrow$ *cur_packet*

on_receive_fine_request

1: send a fine to the source

on_timer$(1/r)$

1: **if** packet i received from P **then**
2: *detected_message(P)*
3: **else**
4: *missed_message(P)*
5: $i \leftarrow i + 1$
6: **if** *is_selfish(P)* **then**
7: report P to source
8: *start_monitor(P)*

Once a parent P is suspected of selfish behavior by LTSM at its child C, the child reports its suspicion to the source node (which is altruistic), and optimistically restarts LTSM. The source then asks the suspected node to pay it a fine. In case the suspected node does not comply, the source asks the suspect's parent to disconnect the non-cooperative node, and the orphan sub-tree reconnects to the spanning tree, bypassing the removed node.

Note that falsely accusing a parent is not beneficial for a child. A false suspicion merely causes the parent to pay a fine, increasing its cost, but with no benefit to the accusing child. Moreover, if the parent fails to pay the fine, the child's utility may decrease due to missed data or reduced performance during the tree reconstruction phase. On the other hand, reporting genuine suspicion is vital to ensure future service. Therefore, adherence to the protocol is a Nash equilibrium.

In a similar way, LTSM can be integrated into mesh-based multicast solutions. For example, EquiCast [14] uses similar threshold-based detection, and fines for readmission. Replacing their selfishness monitor with LTSM automatically makes EquiCast robust to packet loss.

6.2 MANET Routing

A common approach to detecting selfish behavior in a MANET is using reputations. Such reputations are collected by nodes through monitoring the behavior of other nodes. The watchdog mechanisms used in CONFIDANT [4] and later in CORE [19], OCEAN [2] and others, use promiscuous mode to eavesdrop on neighboring nodes and monitor packet forwarding and other network activity.

LTSM can easily be integrated into such mechanisms, seamlessly adding resilience to lossy networks. In the watchdog mechanism, every node keeps track of packets that flow in and out of each of its radio range neighbors. This modus operandi is appropriate for LTSM's tracking of request-based messages. LTSM's *missed_message(N)* method can be called when a packet dropped by node N is detected, while *detected_message(N)* can be called when a packet is correctly forwarded by node N. LTSM can monitor additional request-based traffic, such as route discovery or location queries and replies, as well as constant rate traffic such as keep-alive messages.

7 Conclusions

P2P systems in commercial applications often operate over lossy networks and are bound to experience selfish behavior. We have defined a general service for monitoring and discouraging selfish behavior, which is applicable to a variety of P2P protocols. We have presented such a monitoring service, called LTSM, which is suitable for networks subject to message loss. We have mathematically analyzed LTSM, and shown how to tune its parameters so as to encourage full cooperation, while minimizing the cost for cooperating nodes. Finally, we have shown usage examples of our service for multicast and for MANET routing.

References

[1] Eitan Altman, Arzad A Kherani, Pietro Michiardi, and Refik Molva. Non-cooperative forwarding in ad-hoc networks. In *4th International IFIP-TC6 Networking Conf.*, volume 3462, pages 486–498, 2005.

[2] Sorav Bansal and Mary Baker. Observation-based cooperation enforcement in ad hoc networks. Technical Paper, Stanford University, 2003.

[3] Alberto Blanc, Yi-Kai Liu, and Amin Vahdat. Designing incentives for peer-to-peer routing. In *INFOCOM 2005*, pages 374–385, 2005.

[4] Sonja Buchegger and Jean-Yves Le-Boudec. Performance analysis of the confidant protocol. In *MobiHoc '02*, pages 226–236, 2002.

[5] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *SOSP '03*, pages 298–313, 2003.

[6] Bram Cohen. Incentives build robustness in bittorrent. In *1st Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[7] Jing Dong, Kurt E. Ackermann, Brett Bavar, and Cristina Nita-Rotaru. Mitigating attacks against virtual coordinate based routing in wireless sensor networks. In *WiSec '08*, pages 89–99, 2008.

[8] John R. Douceur. The sybil attack. In *IPTPS '01*, pages 251–260, 2002.

[9] Márk Félegyházi, Jean-Pierre Hubaux, and Levente Buttyán. Nash equilibria of packet forwarding strategies in wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(5):463–476, 2006.

[10] Christos Gkantsidis and Pablo Rodriguez. Network coding for large scale content distribution. In *INFOCOM 2005*, pages 2235–2245, 2005.

[11] Jiangyi Hu. Cooperation in mobile ad hoc networks. Technical report, CS Department, Florida State University, 2005.

[12] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, and Jr James W. O'Toole. Overcast: reliable multicasting with on overlay network. In *OSDI'00*, 2000.

[13] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[14] Idit Keidar, Roie Melamed, and Ariel Orda. Equicast: scalable multicast with selfish users. In *PODC '06*, pages 63–71, 2006.

[15] Lawrence M. Leemis and Kishor S. Trivedi. A comparison of approximate interval estimators for the bernoulli parameter. *The American Statistician*, 50:63–68, 1996.

[16] Harry C. Li, Allen Clement, Mirco Marchetti, Manos Kapritsos, Luke Robison, Lorenzo Alvisi, and Mike Dahlin. FlightPath: Obedience vs choice in cooperative services. In *OSDI'08*, 2008.

[17] Harry C. Li, Allen Clement, Edmund L. Wong, Jeff Napper, Indrajit Roy, Lorenzo Alvisi, and Michael Dahlin. Bar gossip. In *OSDI '06*, pages 191–204, 2006.

[18] Sergio Marti, T J Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00*, pages 116–123, 2000.

[19] Pietro Michiardi and Refik Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, pages 107–121, 2002.

[20] Pietro Michiardi and Refik Molva. A game theoretical approach to evaluate cooperation enforcement mechanisms in mobile ad hoc networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 3–5, 2003.

[21] Tsuen-Wan Ngan, Dan S. Wallach, and Peter Druschel. Incentives-compatible peer-to-peer multicast. In *2nd Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[22] Dimitrios Pendarakis, Sherlia Shi, Dinesh Verma, and Marcel Waldvogel. ALMI: an application level multicast infrastructure. In *USITS'01*, 2001.

[23] Michael Sirivianos, Jong Han Park, Xiaowei Yang, and Stanislaw Jarecki. Dandelion: Cooperative content distribution with robust incentives. In *USENIX*, pages 157–170, 2007.

[24] Vikram Srinivasan, Pavan Nuggehalli, Carla-Fabiana Chiasserini, and Ramesh R. Rao. An analytical approach to the study of cooperation in wireless ad hoc networks. *IEEE Transactions on Wireless Communications*, 4(2):722–733, 2005.

[25] Eric W. Weisstein. Mathworld–a wolfram web resource. wolfram research, inc. http://mathworld.wolfram.com, 1998-2007.

[26] Wei Yu and K. J. Ray Liu. Game theoretic analysis of cooperation stimulation and security in autonomous mobile ad hoc networks. *IEEE Trans. Mob. Comput.*, 6(5):507–521, 2007.

[27] Wei Yu and K. J. Ray Liu. Secure cooperation in autonomous mobile ad-hoc networks under noise and imperfect monitoring: A game-theoretic approach. *IEEE Transactions on Information Forensics and Security*, 3(2):317–330, 2008.

A Formal Proofs

A.1 Proof of Lemma 1

We now prove several auxiliary claims, which are used to prove Lemma 1. The first claim shows that the fine needs to exceed a certain bound, captured by the following function:

Definition 1 (Lower Bound on Fine)**.** *For all natural* n, $0 < n \leq W$*:*

$$f(\tau, W, n) \triangleq \frac{n}{D(W-n,\tau) - \epsilon(\tau)}.$$

Claim 1. *For all natural* n, $0 < n \leq W$*:* $Ecost(W-n,\tau) > Ecost(W,\tau)$ *if and only if* $fine > f(\tau, W, n)$.

Proof. From (2) we have: $Ecost(W,\tau) = W + fine \times \epsilon(\tau)$, and $Ecost(W-n,\tau)] = W - n + fine \times D(W-n,\tau)$. Solving $Ecost(W-n,\tau) > Ecost(W,\tau)$ and isolating $fine$ we get the desired result. □

Therefore, we are looking for a fine so that $fine > f(\tau, W, n)$ for all natural n, $0 < n \leq W$. We first show that if a rational node sends less than τ messages, then it sends no messages at all.

Claim 2. *For all natural n, $W-\tau < n \leq W$: $Ecost(W-n,\tau) > Ecost(W,\tau)$ if and only if $fine > \frac{W}{1-\epsilon(\tau)}$.*

Proof. The expected cost for all $X \in [0,\tau)$, which is $X + fine$, is minimized at $X = 0$. Therefore, the proof follows from Claim 1 with $n = W$. □

Having resolved the case that less than τ packets are sent, we now restrict our attention to the case that τ to $W-1$ packets are sent, i.e., n is in $(0, W-\tau]$. We next simplify the expression for $f(\tau, W, n)$ in order to find a constraint that enforces full cooperation.

Claim 3. *For all natural m, $\tau \leq m < W$: $D(m,\tau) = D(m+1,\tau) + qP(y(m) = \tau)$.*

Proof. We use the following transformation of the regularized incomplete beta function [25] using the Gamma function Γ:

$$I_z(a+1,b) = I_z(a,b) - \frac{\Gamma(a+b)}{\Gamma(a+1)\Gamma(b)}(1-z)^b z^a.$$

From (1), we have $D(m+1,\tau) = I_p(m-\tau+1,\tau+1)$. Therefore,

$$D(m+1,\tau) = I_p(m-\tau,\tau+1) - \frac{\Gamma(m+1)}{\Gamma(m-\tau+1)\Gamma(\tau+1)} q^{\tau+1} p^{m-\tau}$$
$$= D(m,\tau) - q\frac{m!}{(m-\tau)!\tau!} q^\tau p^{m-\tau} = D(m,\tau) - qP(y(m) = \tau). \quad \square$$

The following claim shows that $f(\tau, W, n)$ (Definition 1) is one over the average of $P(y(W-k) = \tau)$ over $k \in [1,n]$, multiplied by q. Recall that $W-n$ here is the number of packets sent.

Claim 4. *For all natural $0 < n \leq W - \tau$:*

$$f(\tau, W, n) = \frac{n}{q\sum_{k=1}^{n} P(y(W-k) = \tau)} \ .$$

Proof. We start by iteratively applying Claim 3 on $D(W-n,\tau)$: $D(W-n,\tau) = D(W-n+1,\tau) + qP(y(W-n) = \tau) = D(W,\tau) + q\sum_{k=1}^{n} P(y(W-k) = \tau)$. Recall that $\epsilon(\tau) = D(W,\tau)$. The result follows by substituting $D(W-n,\tau)$ into the definition of $f(\tau, W, n)$. □

We thus have an expression for $f(\tau, W, n)$, which is a lower bound for the fine (see Claim 1). To ensure full cooperation the bound has to hold for all n. We therefore seek the maximum value of $f(\tau, W, n)$. We start with an auxiliary claim.

Claim 5. *Let $a = \{a_k\}_{k=1}^m$ be a series that increases for $k < A$ and decreases for $k > A$ for some real number A. Let $z = \{z_n = \frac{1}{n}\sum_{k=1}^{n} a_k\}_{n=1}^m$ be the series of partial averages of a. Then the minimum of the series $\{z_n\}_{n=1}^m$ occurs at $n = 1$ or $n = m$. That is, $\min\{z_n\}_{n=1}^m = \min(z_1, z_m)$.*

Proof. Observe that for all natural n, $1 \leq n < m$, $z_{n+1} > z_n$ if and only if $a_{n+1} > z_n$. Since a increases for $k < A$, z also increases for $n < A$. Although a decreases for $k > A$, z may continue increasing for some $n > A$, as long as $a_{n+1} > z_n$. If there is no n, $1 \leq n < m$, such that $a_{n+1} \leq z_n$, then z is always increasing and $\min\{z_n\}_{n=1}^m = z_1$. Otherwise, let j be the smallest index $1 \leq j < m$ so that $a_{j+1} \leq z_j$. Then z is increasing for $k < j$. It is easy to see, by induction on $k \geq j$, that z is decreasing for $k \geq j$. Thus, $\min\{z_n\}_{n=1}^m = min(z_1, z_m)$. □

Claim 6. *The maximal value of $f(\tau, W, n)$ for $0 < n \leq W - \tau$ occurs at $n = 1$ or $n = W - \tau$. That is, $\max\{f(\tau, W, n) | 0 < n \leq W - \tau\} = \max\{f(\tau, W, n) | n = 1, W - \tau\}$.*

Proof. Since y(X) is a binomial random variable, for all natural k, $0 < k < W - \tau$:

$$\frac{P(y(W-k)=\tau)}{P(y(W-(k+1))=\tau)} = \frac{\frac{(W-k)!}{\tau!(W-k-\tau)!} \times q^{\tau} p^{W-k-\tau}}{\frac{(W-k-1)!}{\tau!(W-k-1-\tau)!} \times q^{\tau} p^{W-k-1-\tau}} = \frac{W-k}{W-k-\tau} \times p.$$

Hence, $P(y(W-k)=\tau)$ is increasing for $k < W-\tau/q$, and decreasing for $k > W-\tau/q$. By Claim 5, the average, $\frac{1}{n}\sum_{k=1}^{n} P(y(W-k)=\tau)$, is minimized at one of the extremities ($n=1$, $n=W-\tau$). Hence, the maximum of $f(\tau, W, n)$, which is one over the above average divided by q (see Claim 4), also occurs at either $n=1$ or $n=W-\tau$. □

We are now ready to prove Lemma 1, which is restated below.

Lemma 1 (restated). *For all natural n, $0 < n \leq W$: $Ecost(W-n,\tau) > Ecost(W,\tau)$ if and only if the following constraint holds:*

$$fine > \max\left(\frac{W}{1-\epsilon(\tau)}, \frac{1}{qP(y(W-1)=\tau)}\right).$$

Proof. By Claim 2, for all natural n such that $W-\tau < n \leq W$: $Ecost(W-n,\tau) > Ecost(W,\tau)$ if and only if $fine > \frac{W}{1-\epsilon(\tau)}$. For $0 < n \leq W-\tau$, by Definition 1 and Claim 4:

$$f(\tau, W, n=W-\tau) = \frac{W-\tau}{P(y(\tau)\leq\tau)-\epsilon(\tau)} = \frac{W-\tau}{1-\epsilon(\tau)},$$
$$f(\tau, W, n=1) = \frac{1}{qP(y(W-1)=\tau)}.$$

By Claim 6, the constraint for $0 < n \leq W-\tau$ is the the maximum of the above two values. Hence, the constraint for $0 < n \leq W$ is given by:

$$\max\left(\frac{W-\tau}{1-\epsilon(\tau)}, \frac{1}{qP(y(W-1)=\tau)}, \frac{W}{1-\epsilon(\tau)}\right).$$ □

A.2 Proof of Lemma 2

In order to prove Lemma 2, we first prove three auxiliary claims. The following function, f_{max}, captures the maximum value of f for a given false positive probability δ:

$$f_{max}(\delta, W) \triangleq \max_{1\leq n\leq W} f(\epsilon^{-1}(\delta), W, n).$$

We start by looking for an approximation for $\epsilon^{-1}(\delta)$ (for a large W), using the error function, erf [25].

Claim 7. *Let $\epsilon(\tau)=\delta$. Then for a large W, $\tau \approx qW - 0.5 - \sqrt{2pqW}(\mathrm{erf}^{-1}(1-2\delta))$.*

Proof. For a large W, we can use the de Moivre-Laplace Theorem [25]:

$$\delta = P(y(W) > \tau) = \Phi\left(\frac{\tau+0.5-\mu}{\sigma}\right) + O\left(\frac{1}{\sqrt{W}}\right)$$
$$= \frac{1}{2}\left[1+\mathrm{erf}\left(\frac{\tau+0.5-\mu}{\sqrt{2}\sigma}\right)\right] + O\left(\frac{1}{\sqrt{W}}\right),$$

where Φ is the standard normal cumulative distribution function, $\mu = Wq$, and $\sigma^2 = Wpq$. Hence, $\tau = \epsilon^{-1}(\delta) \approx \mu - 0.5 - \sigma\sqrt{2}\mathrm{erf}^{-1}(1-2\delta)$. □

Next, we approximate $P(y(W-1)=\tau)$, which appears in the constraint of Lemma 1.

Claim 8. *For a large W, if* $\epsilon(\tau)=\delta$, *then* $P(y(W-1)=\tau) \approx (1/\sqrt{2\pi(W-1)pq}) \times \exp(-(\mathrm{erf}^{-1}(1-2\delta))^2)$.

Proof. For a large W, using the de Moivre-Laplace Theorem with $\mu=(W-1)q$, and $\sigma^2=(W-1)pq$:

$$P(y(W-1)=\tau)=\binom{W-1}{\tau}q^{\tau}p^{(W-1-\tau)} \approx \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(\tau-\mu)^2}{2\sigma^2}\right).$$

Using the approximation of τ from Claim 7, we get:

$$\begin{aligned} P(y(W-1)=\tau) &\approx \frac{1}{\sqrt{2\pi(W-1)pq}}\exp\left[-\left[\frac{-\frac{1}{2}-\sqrt{2Wpq}\mathrm{erf}^{-1}(1-2\delta)+q}{\sqrt{2(W-1)pq}}\right]^2\right] \\ &\approx \frac{1}{\sqrt{2\pi(W-1)pq}}\exp[-(\mathrm{erf}^{-1}(1-2\delta))^2]. \end{aligned}$$

□

Claim 9. $\epsilon(0)=p^W$ *and* $f_{max}(\epsilon(0),W)=1/(p^{W-1}q)$.

Proof. When $\tau=0$, a false suspicion occurs if and only if all W messages are lost, therefore $\epsilon(0)=p^W$. Let $\delta=\epsilon(\tau)$. By Lemma 1,

$$f_{max}(\delta,W)=\max\left(\frac{W}{1-\delta},\frac{1}{qP(y(W-1)=\epsilon^{-1}(\delta))}\right)=\max\left(\frac{W}{1-p^W},\frac{1}{qp^{W-1}}\right).$$

It remains to show that $\frac{W}{1-p^W}\leq\frac{1}{qp^{W-1}}$, i.e., that $W\leq\frac{1-p^W}{qp^{W-1}}$. We prove this by induction. For $W=1$, both sides of the inequality are equal. Assume that the inequality holds for $W=k$, $k\in\mathbb{N}$. We have to prove that the inequality still holds for $W=k+1$. To this end, we subtract the inequality for $W=k$ from the inequality for $W=k+1$, and get:

$$1=k+1-k\leq\frac{1-p^{k+1}}{qp^k}-\frac{1-p^k}{qp^{k-1}}=\frac{1-p^{k+1}-p+p^{k+1}}{qp^k}=\frac{1}{p^k},$$

which holds for all $0<p\leq 1$ and all k. □

We are now ready to prove Lemma 2, which is restated below.

Lemma 2 (restated). *For a large enough W, under the constraint of Lemma 1, the expected cost for a cooperative node is minimized at* $\tau=0$. *The minimal expected cost is* $W+p/q$, *and the fine has to satisfy* $fine>1/(qp^{W-1})$.

Proof. The expected cost for a cooperative node is $W+fine\times\epsilon(\tau)$. Let $\delta=\epsilon(\tau)$. We start by looking at the second term of the constraint in Lemma 1. Let

$$g(\delta,W)\triangleq\frac{1}{qP(y(W-1)=\epsilon^{-1}(\delta))}.$$

By Claim 8, for a large W, the asymptotic behavior of $g(\delta,W)\times\delta$ (for constant W, p, and q) is:

$$g(\delta,W)\times\delta\propto\frac{\delta}{\exp(-(\mathrm{erf}^{-1}(1-2\delta))^2)},$$

which is increasing in δ. Hence, the minimum of $g(\delta,W)\times\delta$ is obtained at the minimal $\delta=p^W$, where $g(\delta=p^W,W)\times\delta=\frac{\delta}{p^{W-1}q}=\frac{p}{q}$.

Now look at the first term of the constraint in Lemma 1.
Let $h(\delta, W) \triangleq W/(1-\delta)$. Notice that $h(\delta, W) \times \delta = (\delta \times W)/(1-\delta)$ is also an increasing function in δ, with a minimum at $\delta = p^W$.

By Claim 9, we get:

$$fine > \max\{g(p^W, W), h(p^W, W)\} = f_{max}(p^W, W) = \frac{1}{p^{W-1}q} \ . \qquad \square$$

A.3 Proof of Lemma 3

For the sake of clarity, we first restate Lemma 3:

Lemma 3 (restated). *Given $fine$ and W such that W is large, if $fine > W/(1-\delta_{min})$, and $\delta_{min} < 0.5$, then choosing $\tau = \lfloor \tau_{min} + 1 \rfloor$ warrants full cooperation and yields a minimal expected cost over all possible values of τ.*

Proof. Let $\delta = \epsilon(\tau)$. Recall that the expected cost for a cooperative node is equal to $W + \delta \times fine$. As W and $fine$ are given constants, the expected cost is minimized, while warranting full cooperation, by finding the smallest δ that satisfies the constraint of Lemma 1:

$$fine > f_{max}(\delta, W) = \max\left(\frac{W}{1-\delta}, \frac{1}{qP(y(W-1) = \epsilon^{-1}(\delta))}\right)$$

The lemma's preconditions guarantee the first constraint, which is satisfied for all $\delta < 1 - W/fine$. Using Claim 8, the second constraint becomes:

$$fine > \frac{\sqrt{2\pi(W-1)pq}}{q\exp(-(\mathrm{erf}^{-1}(1-2\delta))^2)},$$

which is decreasing in δ for $\delta < 0.5$. Thus, from the second constraint, with $\delta < 0.5$, we get:

$$\delta > \delta_{min} = \frac{1 - \mathrm{erf}\left(\sqrt{-\ln\left(\frac{\sqrt{2\pi(W-1)pq}}{q \times fine}\right)}\right)}{2}$$

The result for τ_{min} is attained by substituting δ_{min} into the expression for τ given in Claim 7. $\square$