Computer Networks 75 (2014) 58–74

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Revisiting security of proportional fair scheduler in wireless cellular networks

Hanjin Park [a], Yung Yi [b,*], Yongdae Kim [b]

[a] Department of Computer Science, KAIST, Daejeon, South Korea
[b] Department of Electrical Engineering, KAIST, Daejeon, South Korea

ARTICLE INFO

Article history:
Received 11 March 2014
Received in revised form 5 August 2014
Accepted 15 September 2014
Available online 6 October 2014

Keywords:
Proportional fair scheduler
Adaptive modulation and coding (AMC)
Hybrid automatic repeat request (HARQ)
Outer loop link adaptation (OLLA)
Channel quality indicator (CQI)
Feedback fabrication

ABSTRACT

Proportional fair scheduling (PFS) and its variants have been widely deployed in 3G/4G systems, where base stations (BSs) use channel quality indicator (CQI) from users as a channel feedback. Series of recent papers have shown that PFS may not perform as expected, when these client-side feedbacks such as CQI or NACK are fabricated. These results, however, were obtained without considering all relevant components in practice that are designed to handle various corner cases. In this paper, we revisit the impact of CQI and NACK fabrications, and also the fabrication of ACK (which has been largely ignored in prior work), by jointly considering all known components such as adaptive modulation and coding (AMC), hybrid automatic repeat request (HARQ), outer loop link adaptation (OLLA) and PFS as a whole. To consider many practical fabrication scenarios, we study both cases when only a single feedback and a smart combinations of multi-feedbacks are exploited for selfish and/or malicious purposes. From these studies, we draw in-depth findings with large practical implications, most of which are in sharp contrast to those in prior work on the security of PFS.

1. Introduction

1.1. Motivation

In mobile cellular networks, channel conditions are time-varying and user/location-dependent. This multi-user diversity has been exploited to achieve high performance in scheduling users, referred to as *opportunistic* scheduling, which preferably serves users with better channel conditions. Thus, most opportunistic schedulers naturally need users' cooperation, i.e., periodically sending their feedbacks to BS on channel quality and frame reception failures or success, in the form of CQI (channel quality indicator) and ACK/NACK. An opportunistic scheduler that has been extensively studied in the literature, is PFS (proportional fair scheduler) [1] which is provably optimal in throughput and fairness, e.g., [2–4,1]. In practice, PFS or its variants are recommended by the current 3G standards, such as HSDPA [5] and EV-DO [6,7] as well as the 4G standards such as LTE [8] and Mobile WiMAX [9].

However, users are not always cooperative. There may be users who can report "wrong" feedbacks in various ways. First, the users can inflate CQI via fabricating communication chipset or the communication-related kernel code with the goal of receiving more service rate than the fair share (selfish objective) or blocking other users' service (malicious objective).[1] In most mobile terminals, the CQI report module is implemented in the baseband

* Corresponding author. Tel.: +82 42 350 3486.
E-mail addresses: hjpark@lanada.kaist.ac.kr (H. Park), yiyung@kaist.edu (Y. Yi), yongdaek@kaist.ac.kr (Y. Kim).

[1] Being malicious or selfish just follows a fabricator's intention of cheating. However, it is possible that a selfish user's behavior can be indirectly malicious because her selfish action can negatively impact other normal users. In this paper, we assume that selfish users' main interest is the increase of its own (long-term average) throughput whereas malicious users have interests of blocking other normal users as long as possible.

http://dx.doi.org/10.1016/j.comnet.2014.09.009
1389-1286/© 2014 Elsevier B.V. All rights reserved.

communication chipset. Although it is beyond this paper's scope to discuss how difficult it is to fabricate a chipset, it does not seem entirely impossible. Also, recent advances in software-defined radio technology are expected to enable such a personal fabrication, e.g., see [10] for a project implementing a software based 2.5G GSM. Second, a cellphone maker may intentionally include the channel feedback inflation module in order to win a larger market share, by letting users experience higher rates than those using the cellphones from other makers. This does not seem impossible from the examples of the WiFi market, where a few WiFi NIC cards are observed to deviate from the standard backoff mechanisms [11]. Third, the users can fabricate NACK via cheating communication chipset to generate lots of retransmissions to block other users service such as Denial of Service (DoS) attack for malicious objective. Recent studies report such security concerns of PFS, e.g., malicious and selfish impacts of CQI fabrication [12–16], and NACK attack for a malicious purpose [17], or their combinations [18] (see Section 3.4 for their more details and comparison to our work).

Despite extensive afore-mentioned research efforts, PFS's security has yet to be studied more, and we revisit such an issue in this paper. As more details will be discussed in Section 3.4, prior work on PFS security has often been conducted under impractical setups, e.g., without considering a variety of relevant components in an integrated manner or with simply ignoring the current implementation practice. In this paper, we jointly consider (i) adaptive modulation and coding (AMC), (ii) hybrid automatic repeat request (HARQ), (iii) outer loop link adaptation (OLLA) and (iv) PFS as a whole, and examine the complex inter-plays among them to uncover how robust PFS is to CQI and ACK/NACK feedback fabrication. To briefly summarize each component, AMC, also called ILLA (Inner Loop Link Adaptation) [6,7,5], adaptively sets the modulation and the coding parameters to match scheduled user's channel condition, providing the highest link rate with the maximum allowable BLER (Block Error Rate) (typically 10%). HARQ (hybrid automatic repeat request) [19–22] is an error control technique mixing forward error correction (FEC) and ARQ to combat against uncertain wireless fading. OLLA at BSs fights against imperfect CQI feedback by adaptively correcting CQI using past ACK/NACK information to meet target error threshold [23–29].

In particular, we note the current practice in (i) AMC (or ILLA) working under a *static* BLER threshold mainly for implementation simplicity (typically 10%), and (ii) OLLA that is a calibration mechanism at the BS side for imperfect CQI reports, both of which have rarely been considered in existing studies on PFS's security. Since OLLA at BS adjusts CQI values to meet a target BLER using ACK/NACK histories, OLLA may detect fake CQI feedbacks and correct it. However, we cannot ensure OLLA's capability of defending against CQI/ACK/NACK's fabrication, because (a) it takes time that OLLA accumulates ACK/NACK histories to alarm fabrication attacks, and (b) since OLLA fully trusts ACK/NACK feedback from users, ACK/NACK fabrication or their combinations may screw up OLLA. This motivates us to revisit the security of PFS over more practical setup and find out how PFS is robust to selfish and malicious fabrications.

1.2. Main contributions

We summarize our major findings and contributions of this paper here.

(1) Robust to single fabrications. First, we study the impact of fabrications that use only a single feedback, i.e., CQI, ACK, or NACK only.

(a) *CQI fabrication.* OLLA's role lies in "correcting" the fabricated CQIs to meet the target BLER (typically 10%). Thus, it turns out to be difficult for selfish or malicious fabricators to increase their throughput, or change PFS scheduling priority via CQI fabrication. This is a stark contrast to prior works [12,14–16,33,34] which have largely ignored OLLA on top of static BLER threshold in their analysis (see Section 4.2).

(b) *NACK fabrication.* NACK fabrication, which pretends to have frame failures and exhaustively sends NACK to a BS, may starve other normal users due to high priority of retransmissions, thus they are mainly used for malicious purposes. Such a malicious effect has been reported by [17]. However, our findings tell us that OLLA is capable of mitigating NACK fabrications to some extent, since OLLA reduces fabricators' CQI as a kind of penalty for incoming NACKs for a certain time duration. Using these findings, we also propose a defense mechanism which is much simpler that in [17].

(c) *ACK fabrication.* ACK fabrication, which pretends to receive frames successfully even upon errors, enables OLLA to increase the fabricator's CQI because OLLA misjudges that the fabricator's channel condition is good. Seemingly, this may lead to the increase of fabricators' throughput, but we find that such selfish gain has no actual effects, since inflated CQI due to ACK fabrication results in many frame decoding errors. More interesting selfish effects of ACK fabrication are discovered when combined with other multi-feedback fabrications, as described next.

(2) Robust to joint fabrications. Next, we consider smarter fabrications that jointly use multi-feedbacks; In particular, we study the following two joint fabrication approaches:

(a) *Fake NACK + minimum CQI.* In this fabrication, a malicious user reports the minimum CQI when it starts to fabricate NACKs. This has the effect of nullifying OLLA's penalty operation because OLLA cannot decrease the minimum CQI further. This joint fabrication leads to a burst of scheduling changes to the malicious user, and thus blocking other normal users. Our finding is in contrast to that in [18] that only NACK fabrication as is the most effective among combinations of CQI and NACK fabrications in terms of malicious effect.

(b) *CQI inflation + fake ACK.* As mentioned earlier, just fabricating CQI does not help significantly. However, in fabricating CQIs, a selfish user can sporadically use fake ACKs to weaken OLLA's penalization by letting OLLA misunderstand that the current MCS selection is appropriate for the given channel condition (of the selfish user). It turns out that this joint fabrication scheme can increase link-level throughput. However, it cannot significantly increase application-level throughput using loss-sensitive transport protocols (e.g., TCP) due to the throughput decrease led by intermittent frame errors.

The rest of the paper is organized as follows. Section 2 provides the backgrounds on PFS and its relevant components such as ILLA, OLLA, and HARQ. Section 3 investigates the issues of fabricating CQI, ACK, and NACK. In Section 4, we perform our simulation based studies of single fabrications with focus on coupling with OLLA. In Section 5, we consider two smarter fabrications that jointly use a multiple of feedback fabrications. In Section 6, we propose the countermeasures for some fabrication schemes, and conclude in Section 7.

2. Background

2.1. Link adaptation

The role of link adaptation lies in adaptively adjusting the link data rate that has the best match with mobile terminals' channel condition, while being robust to errors. Two types of link adaptation mechanism are popular in practice: Inner-loop and outer-loop ones.

2.1.1. ILLA (inner loop link adaptation)

ILLA, also called AMC (adaptive modulation and coding), maps the estimated SNR to a data rate by choosing a combination of modulation and coding schemes (MCS) (e.g., BPSK and coding rate 1/2) out of a finite set of available MCS levels. Available MCS levels are specified by a cellular standard. An MCS for the estimated SNR is chosen such that it provides the largest data rate while satisfying a BLER bound $\bar{e}$. The bound $\bar{e}$ is a system parameter, where, for example, in LTE $\bar{e} = 0.1$ [27]. As a simple example, consider the case of only two MCS levels, as shown in Fig. 1. Associated with each MCS level is a SNR-BLER curve. For instance, when a user estimates SNR h, it selects the first MCS level (and thus the corresponding modulation and coding schemes), because the second MCS level will generate a larger BLER than $\bar{e}$, and sends its index to its associating BS as a CQI. We note that even for a given combination of MCS, its actual transmission rate may differ depending on other system parameters and other phy-layer technologies, e.g., operating frequency bands. For example, a combination of QPSK with rate 1/2 provides 5040 kbits/s in WiMAX [9], but 731 kbits/s in HSDPA Category 10 device [30]. BS collects the CQI feedbacks from all mobile terminals in its cell, and set a transmission rate to a scheduled user according to the user's CQI feedback or after some calibration procedure, called OLLA, as explained next.

2.1.2. OLLA (outer loop link adaptation)

ILLA does not suffice for link adaptation for the following so-called CQI offset problems: first, there exists time

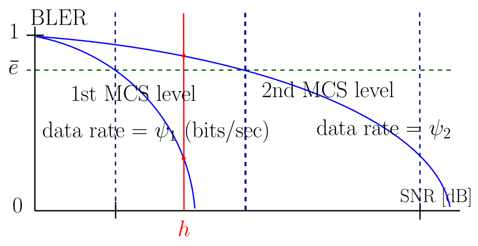


Fig. 1. Example of ILLA with 2 MCS levels. The first and second MCS levels support data rate ψ_1 (bits/s) and ψ_2, respectively. When a user's channel is h (dB) in SNR, the user selects the first MCS level (and thus the corresponding modulation and coding schemes), because the second MCS level would generate a larger BLER than the threshold $\bar{e}$.

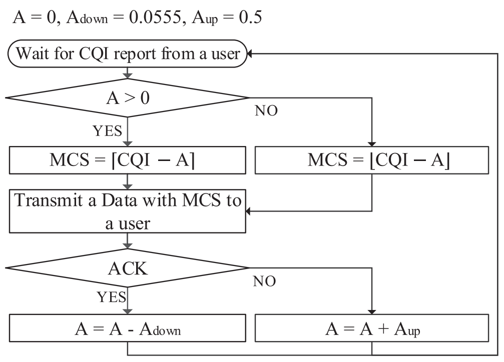

A = 0, Adown = 0.0555, Aup = 0.5
Wait for CQI report from a user
A > 0
YES
NO
MCS = ⌈CQI − A⌉
MCS = ⌊CQI − A⌋
Transmit a Data with MCS to a user
ACK
YES
NO
A = A - Adown
A = A + Aup

Fig. 2. Flow chart of OLLA (outer loop link adaptation) in [24].

difference between users' channel estimation and BSs' actual transmission, and thus MCS selection may not be proper, especially for users with fast channel variation. Second, a malfunctioned mobile phone can continuously send "wrong" (potentially very high) CQI. What OLLA does is that BSs readjust the received CQI to meet the BLER bound by smartly guessing correct CQIs. There exist several OLLA proposals [23,24,27,26,28,29] whose core algorithmic ideas are largely shared: they all use ACK/NACK histories to check whether the earlier MCS level selections have been appropriate or not.[2]

We now briefly explain how OLLA works using the algorithm in [24], as described in Fig. 2. A BS maintains ACK/NACK history of each user, and also has a per-user variable A, which decrease (resp. increase) by A_{down} (resp. A_{up}) for each ACK (resp. NACK). The fact that $A < 0$ means that the earlier MCS selections were good enough for the user to receive data, whereas $A > 0$ (i.e., many NACKs) means that actual channel quality of the user is bad, compared to the CQI feedback. Thus, once receiving the per-user CQI feedback as an MCS index, the BS regards CQI $-$ A (rather than the user's pure CQI feedback) as the real CQI feedback, and chooses the MCS index $\lceil \text{CQI} - A \rceil$ when $A > 0$, and when $A < 0$, chooses the MCS index $\lfloor \text{CQI} + A \rfloor$.

[2] The major difference among the various OLLA algorithms in literature comes from the parameters in OLLA, such as the number of frames to inspect to check whether the measured BLER exceeds the BLER bound or not and the number of frames needed to lower or lift up the MCS levels.

According to this algorithm, as discussed in [24], it is easy to see that the converging BLER for given A_{up} and A_{down} becomes $BLER = 1/\left(1+\frac{A_{\mathrm{up}}}{A_{\mathrm{down}}}\right)$, and to meet the target BLER 10%, A_{up} and A_{down} can be suitably chosen. Note that there are many combinations of A_{up} and A_{down} each of which leads to BLER to be 10%. Even one can propose a dynamic way of adjusting those two parameters, but this paper considers a fixed choice of $A_{\mathrm{down}} = 0.0555$ and $A_{\mathrm{up}} = 0.5$, as suggested by [24] due to the following reasons: (i) channel status is highly uncertain and heterogeneous across users, and (ii) implementation should be simple.

2.2. HARQ (Hybrid ARQ)

The link adaptation guarantees BLER only less than $\bar{e}$. Hybrid ARQ (HARQ) recovers the bit errors by combining ARQ and forward error correction (FEC). In HARQ, if the received coded data fails to pass error detection (ED) (e.g., cyclic redundancy check (CRC)), each user sends retransmission requests (to the BS) within a pre-defined maximum number of times, e.g., 3 in WiMAX. There are two types of HARQ: (i) Type-I HARQ adds both error detection (ED) and forward error correction code (FEC) to each frame, (ii) Type-II HARQ only adds ED to each frame when the frame is transmitted for the first time, and if the frame includes error, the frame is retransmitted with both ED and FEC. Thus, Type-II uses less channel bandwidth than Type-I, especially, when frame errors rarely occur. Different from the "pure ARQ", HARQ stores incorrectly received coded data and combines it with the newly received data via retransmission. In this paper, we focus on Type-I HARQ because it is just practically preferred than Type-II HARQ due to implementation simplicity, but more importantly, our analysis does not show significant difference between those two types, because CQI inflation results in frequent retransmissions and both types turn out to add ED and FEC information to the retransmitted frame, i.e., consuming similar bandwidths. We refer the readers to [19–21] for more details. In conjunction with link adaptation, HARQ is also widely employed in HSDPA, LTE, Mobile WIMAX, and their evolutions.

2.3. User scheduling: PFS (proportional fair scheduler)

2.3.1. Basic Algorithm

A BS changes the scheduled user over time slots, where a time slot length is appropriately selected by each standard based on delay spread and inter-symbol interference. On each time slot t, a BS receives a CQI from its associating users, adjusts CQI and maintains their instantaneous data rates, $r_u(t)$, for each user u, resulted from link adaptation schemes. In PFS, the BS finally selects a user $u^\star(t)$ to schedule as follows:

$$u^\star(t) \in \arg\max_{u\in U} \frac{r_u(t)}{R_u(t-1)}, \tag{1}$$

where U is the set of users and $R_u(t)$ is the *averaged* service rate to the user u up-to time t, updated by:

$$R_u(t) = \beta r_u(t)\mathbf{1}_u(t) + (1-\beta)R_u(t-1), \tag{2}$$

where $0 < \beta < 1$ is a weight constant, and $\mathbf{1}_u(t)$ is an indicator function, where $\mathbf{1}_u(t) = 1$, if the user u is scheduled and 0 otherwise. The rationale behind PFS is that by giving higher preference to the user with *better channel* condition and *less aggregate* service, a certain fairness (formally proportional fairness) can be guaranteed [1–4].

2.3.2. HARQ retransmission and R_u update rule

We finally discuss the retransmission issues in HARQ and its inter-play with PFS, and present the system assumed in this paper. First, we assume that each retransmission request gets higher priority than other normal frames, ignoring the scheduling order of PFS mainly for small delay and high throughput (see [16] for the throughput increase due to such a priority-based scheduling). However, note that in practice there exists a maximum retransmission limit, e.g., 3 in LTE. Thus, after the limit, the retransmission frame should compete with other normal frames under PFS. Second, we assume that BSs update $R_u(t)$ only for successfully transmitted frames (i.e., the acked frames), as in most of the related works [1,2,17,12,13,18].

3. Feedback fabrications: issues and questions

BSs employ three feedbacks (CQI, ACK, and NACK) from users, which can be fabricated for both selfish and malicious purposes. In this paper, we mainly focus on the selfish features of CQI and ACK and on the malicious features of CQI and NACK for the following reasons: To take malicious effects, a malicious user should be able to monopolize BS's service time and starve other users for some time duration. ACK fabrication is not malicious by nature because just pretending to receive frames well does not increase the fabricator's service chances. Different from ACK, in NACK fabrication, reporting a series of NACKs temporarily stops a BS from working in a normal mode and allows the BS to give higher priority to the NACK fabricator due to the temporal priority change for retransmissions. We now discuss more details of the issues and the questions of fabrications made by three feedbacks.

3.1. CQI fabrication

As explained in the earlier sections, CQI is a critical feedback used in AMC-based link rate adaptation and then the resulting adapted rate is exploited in PFS. If AMC is assumed "perfect" in the sense that the provided rates are almost continuous (i.e., a significantly fine granularity of modulation and coding schemes), it is expected that CQI fabrication (i.e., inflating CQIs) always hurts the fabricating users, because the fabrication immediately leads to a sub-optimal rate selection. However, in practice, there exists only a *restricted, discrete* rates available (e.g., 16 in WiMAX) in conjunction with a *static* BLER threshold (10% in many systems) for implementation simplicity, which opens possibility of obtaining a larger share of throughput (than a proportionally fair share). However, we also have OLLA, which detects users' fabricated CQIs (e.g., by too many frame errors) and penalizes them to meet the static BLER threshold.

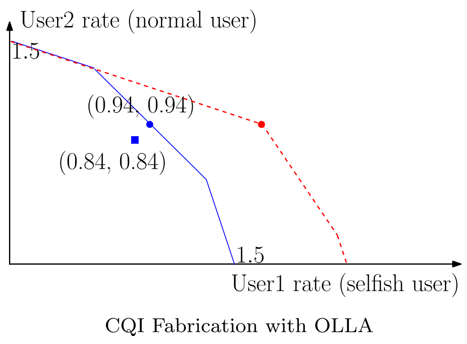


Fig. 3. Blue solid line: BS-provided rate region. Red dashed line: BS-provided rate region under CQI inflation. (0.94, 0.94) are the proportional-fair rates that the BS provides, but the users experience actual throughput (0.84, 0.84). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We illustrate such complex inter-plays using a simple example, where we consider a BS with two associating users, and only two MCS levels exist, providing 3 Mbps and 1 Mbps data rates, respectively. Assume that each user follows the same i.i.d. channel process, experiencing 3 Mbps MCS with probability 0.25 and 1 Mbps MCS with probability 0.75, simply denoted by [0.25, 0.75]. For ease of exposition, we introduce a useful notion of *rate region*, which is the set of all achievable long-term rate vectors, popularly used in theoretical analysis of PFS [1–4]. In our example, the rate region is given in the blue curve of Fig. 3.[3] In terms of the rate region, what PFS does is to provide the long-term rates for users, where in our example, (0.94, 0.94) is the proportionally fair rates for two users. Note that this rate region is only what is from the BS's perspective. A user may experience a different (typically lower) rate, because of throughput reductions generated from frame errors and time waste due to retransmissions. In our example, our simulation shows (0.84, 0.84) user rates.

We now discuss what happens if a user inflates its CQI for selfish or malicious objectives. When a selfish user who wants to increase its long-term average throughput inflates its CQI (this CQI inflation corresponds to allowing the user to apply a larger BLER threshold, say 30%, rather than the system-defined BLER 10%), CQI fabrication from the user lets the BS schedule users and it turns out that a proportional fair point is on a modified (typically larger) rate region, because better channel conditions are reported than the actual ones, see the red dashed line in Fig. 3. Thus, as a positive impact, the rate region is enlarged, but more frame errors and thus larger time waste due to increasing retransmissions may have negative impact on the actual user throughput. On the contrary, since a malicious user wants to block the other normal users for a certain time duration, the malicious user, say u, inflates its CQI such that $\frac{r_u(t)}{R_u(t-1)}$ of (1) (see PFS in Section 2.3) is the highest among all users in a cell at time t slot and monotonically increases its inflated CQI (because $R_u(t-1)$ is increasing due to the update rule of (2)) until the maximum CQI values (e.g., 30 in LTE) to monopolize the PFS for the next consecutive time slots after time slot t. However, if the frame errors caused by inflated CQI of a selfish user or a malicious user persist for a certain time duration, OLLA detects the "wrong" CQIs and penalizes the fabricating user by reducing the CQI value applied to PFS. This complex coupling between discrete MCS levels under a static BLER together and OLLA makes the analysis of CQI fabrication's selfish or malicious effects non-trivial and situation-dependent.

[3] We refer the readers to [32] on how to characterize the rate region.

3.2. ACK fabrication

A user may pretend to receive a frame successfully by reporting an ACK even under error. One of the goals of this ACK fabrication lies in deceiving OLLA into increasing the CQI for PFS. Thus, this fabrication is in effect when the selfish gain from evading OLLA exceeds the throughput loss due to 'intentional' frame losses. It seems that an upper-layer protocol which is highly sensitive to frame errors, e.g., TCP, does not have much selfish gain, whereas loss-insensitive applications, e.g., real-time video flows may have some selfish effect. However, the effect of this fabrication should clearly depend on many other factors such as channel conditions and also other fabrications, e.g., CQI fabrication. We will later study the impact of ACK only fabrication as well as joint fabrication, e.g., ACK + CQI.

3.3. NACK fabrication

Each user sends NACK to its BS in order to report the missing frames, where the BS responses to those reports through retransmissions. As mentioned earlier, the retransmitted frames get higher priority up to the given retransmission limit. This temporal priority change can be exploited by malicious users, i.e., the fabricator keeps sending NACKs to prevent other normal users from being served. However, this is far from simple again due to its complex coupling with PFS and OLLA, as explained in what follows: Once a series of fabricated NACKs are fed back to the BS, the NACKs after the retransmission limit will lose high priority, and compete with other normal frames for scheduling chances. In this moment, the NACK fabricator's averaged service rate $R_u(t)$ is relatively smaller than others' because the fabricator pretends to have frame errors by fake NACK reporting. Since PFS assigns high priority to the user with high $\frac{r_u(t)}{R_u(t)}$ value, it is highly likely that the fabricator can be served again by the scheduler. However, a series of NACK fabrications is regarded as an event that launches OLLA into using a smaller CQI than the reported one from the user, which in turn leads to the reduction of the instantaneous rate $r_u(t)$. This results in delay of re-fabrication, because PFS assigns low priority to the users with low $r_u(t)$. This *starvation due to NACK fabrication* is temporarily resolved later after $R_u(t)$ becomes sufficiently small (due to $R_u(t)$'s update rule in (2)), thereby the fabricator obtains a scheduling chance later again. The amount of the impact of a malicious user's NACK fabrication significantly depends on how aggressively OLLA respond to NACK reports and the situation-dependent changes of $R_u(t)$.

3.4. Discussion on prior work

We now present the main difference of our paper from other prior work on feedback fabrication in cellular networks.

3.4.1. CQI fabrication

Selfish objective. Kavitha et al. [33,34] presented a game-theoretic analysis to study the impact of CQI inflation on PFS, coming from the fabricator's intention of increasing its own average throughput, thus selfish. However, selfish gain here turns out to stem from stealing time shares from other normal users, and this tight coupling between being selfish and malicious is due to the model simplification for mathematical tractability, where other relevant components such as AMC and HARQ are highly abstracted. Thus, it is not clear whether this analysis is transferred to practice or not.

Malicious objective. Racic et al. [12,13] studied the impact of malicious users with CQI fabrication on PFS, and showed that with a single malicious user, normal users' service can be stopped for just about 20 consecutive time slots in the 3G (HSDPA) environment. Their focus is more on group CQI fabrication, where multiple colluders cooperate and block normal users' service in an alternative fashion. However, again AMC, HARQ, and OLLA are not explicitly considered, and their analysis is made just from the perspective of time-slot shares. Kim et al. [14] also analyzed CQI fabrication for malicious purpose, where OLLA is still not considered.

To summarize, most prior work [33,34,12–14] on CQI fabrication has been studied under impractical setups, e.g., without various components in an integrated manner or with simply ignoring the current implementation practice. In particular, OLLA, which is a crucial component in the study of PFS's security, is not taken into consideration in all of prior work. Thus, as will be presented throughout in this paper, our study often shows the messages in practice which completely differs from those in prior work.

3.4.2. NACK fabrication

Ben-Porat et al. [17] studied the NACK fabrication's capability of blocking normal users' service, and showed that when a malicious user keeps reporting NACKs to BS, the user can actually monopolize PFS. However, OLLA and the static BLER threshold are completely ignored in [17], so their analysis leads arguably wrong conclusion. However, with their conclusion, they proposed a modified PFS that slightly changes the $R_u(t)$ update rule, and claimed that it solves the malicious attack under fairness. In this paper, we provide the analysis using simulation results that if OLLA is considered, significantly different PFS' behaviors against malicious NACK fabrication from those in [17], and thus a different defense mechanism can be applied.

3.4.3. CQI + NACK fabrication

Pelechrinis et al. [18] studied the impact of joint CQI [12] and NACK fabrications [17] by investigating various combinations of CQI/NACK feedback. They concluded that only NACK fabrication as in [17] is the most effective among combinations in terms of malicious effect. However, our study disagrees to it in part, and a certain degree of combination effect exists, which comes from our integrated way of considering all the relevant components.

4. Analysis of single fabrication schemes

In this section, we perform simulation-based studies of "single" fabrication schemes, i.e., using one feedback only. This analysis of single fabrications also helps us to further study smarter fabrications for better understanding of PFS's robustness. We first start by describing the simulation setup and metrics.

4.1. Environment and metric

In this section, we run extensive simulations under a variety of configurations to analyze the impact of CQI, ACK, and NACK fabrications. Our simulation is based on 3G cellular systems, where we consider a single cell environment interfered by six neighboring BSs. A major difference of our setup from 4G in our context lies in using multi-carriers when the BS schedules a user (thus multiple users can be selected at one time-slot). However, our analysis over a single-carrier system provides valuable insights and findings much of which can also be transferred to multi-carrier systems, because scheduling multiple users may be somewhat orthogonal to how the system responds to the feedback fabrications.

We conduct our simulations at two levels: link-level and application-level. The simulation settings of physical layer and MAC layer at those two levels are the same except whether the transport and application layers are considered or not. We use ns-2 simulator with the patch of EURANE (Enhanced UMTS Radio Access Network Extensions).

Cell radius is set to be 1 km and BS has 10 users with one fabricator and nine normal users which are uniformly located in the cell. A user has 3GPP HSDPA User Equipment (UE) release 5 categories 10 device (maximum download speed 14.4 Mbps, e.g., iPhone 4S) which has 30 CQI levels [35]. For HARQ, we use *Chase Combining* which is a technique combining the retransmitted and the original frames to recover the error-free frames where the retransmission frames are identical copies of the original ones [19,20,22] and Type-I HARQ (for the detail see the Section). For a maximum HARQ retransmission limit, we use 3 as in 3G HSDPA and 4G LTE [27]. For traffic model, we consider CBR traffic with 3.75 ms interval and packet size 512 KB and FTP traffic with TCP NewReno. From the user to BS, we set the parameters as follows. The user initiates a HS-DSCH channel with the BS. The BS is connected to one Radio Network Controller (RNC), which controls the BS operations such as allocating radio resources and handling soft handovers between two cells, using 622 Mbps Iub link. The Iub link's delay is set to 15 ms. The RNC, SGSN (General Packet Radio Service (GPRS) Support Node), GGSN (Gateway GPRS Support Node) are interconnected with 155 Mbps links with 6 ms delay. The target server is connected to GGSN with 100 Mbps link with 8 ms delay. Other parameters are summarized in Table 1.

Table 1
Environment for NS-2 Simulation.

Item	Value
Carrier freq., Channel bandwidth	2 GHz, 5 Mhz
Total tx. power of BS, Antenna gain of BS	38 dBm, 17 dBi
Thermal noise density	−174 dBm/Hz
Frame length, Cell layout	2 ms (TTI), Hexagonal, omni cell
Cell Radius, Number of active users	1 km, 10
HARQ Scheme	Type-I, Chase Combining
Path loss model	ITU-R M.1225 Pedestrian A 3 km/h
Slow fading model	Log-normal dist'n
Std. deviation of slow fading	8.0 dB
Max. num. of retransmissions	3
UE category	10
CQI mapping table	Table 7D in 3GPP TS 25.214 v11.4.0 [36]
User distance from BS	500 m
TCP type, TCP traffic	NewReno, FTP
UDP traffic	CBR
CBR traffic interval, Packet size	3.75 ms, 512 KB

To generate time-varying channels, we use a channel model generator which was included in EURANE package to generate the UMTS channel [31]. In modeling the propagation environment, we use ITU-R M.1225 Pedestrian-A radio channel profile with the 3 km/h velocity path loss model which is frequently used in other cellular simulations [36–38] and log-normal shadowing fading model with a standard deviation 8 dB. Once the transmission powers and all the channel gains are determined, the data rates for users are calculated using the CQI-SNR mapping table [36].

4.1.1. Metric

In the simulation plots, we often use different metrics to present the various impacts of PFS's security. We list up those metrics for readers' convenience in what follows:

- **PoF (Price of Fabrication)**: the ratio of long-term throughput of the fabricator to that of the non-fabricator. A higher PoF implies larger fabrication gain.
- **Penalty counter**: this per-user metric represents the number of CQI steps penalized by OLLA at certain time slot (e.g., when a user reports CQI 5 and OLLA runs, resulting in CQI 3, then the user's penalty counter increases by $2 = 5 - 3$). Note that the penalty counter can be negative when OLLA boosts up the reported CQI.
- **Scheduled slot**: this is also a per-user metric, corresponding to the total number of slots scheduled to a target user.
- **Duration of penalty**: average consecutive time slots penalized by OLLA.
- **Time slot occupancy**: average consecutive time slots occupied by the malicious user.

4.2. CQI fabrication

4.2.1. Selfish objective

We first examine the impact of CQI fabrication for selfish purpose. In particular, we focus on CQI inflation, for which we consider three plausible inflation policies, (a) *BLER-x*, (b) *MERA*, and (c) *MERAQ*, by introduced in [15,16] to increase the average throughput of each transmission, each of which seems to be simply employed by a selfish user.

4.2.1.1. CQI inflation methods.

(a) BLER-x. In this policy, a fabricator selects a MCS index (i.e., CQI) $l^\star$ which has the highest data rate satisfying BLER x%, formally described as:

$$l^\star = \arg \max_{l \in \text{all MCS indexes}} \{\psi_l | f_l(h) < x\}, \quad (3)$$

where ψ_l is the transmission rate when the MCS level l is applied and $f_l(h)$ is the BLER of MCS level l for channel status h. The case $x = 10\%$ corresponds to what has been defined in practice. In our paper, we often use $x = 30\%$ for CQI inflation, unless specified explicitly.

(b) MERA (Max Effective Rate with AMC). In this policy, a fabricator reports the CQI $l^\star$ that maximizes the "effective" received rate of the first transmission without retransmissions. By "effective", we mean that the decoding probability $1 - f_l(h)$ is explicitly considered, formally described as:

$$l^\star = \arg \max_{l \in \text{ all MCS indexes}} \psi_l(1 - f_l(h)). \quad (4)$$

(c) MERAQ (Max Effective Rate with AMC and HARQ). This policy slightly modifies MERA by computing "expected effective" user rate that additionally considers retransmissions, formally stated:

$$l^\star = \arg \max_{l \in \text{ all MCS indexes}} \sum_{k=1}^{\overline{H}} \left(\frac{\psi_l}{k}\right) f_l(h)^{(k-1)} (1 - f_l(h)), \quad (5)$$

where $\overline{H}$ is the retransmission limit of HARQ. In MERAQ, ψ_l/k is the average received rate when kth retransmission succeeds in being decoded over k time slots, and $f_l(h)^{(k-1)}(1 - f_l(h))$ is the probability that the (re) transmission becomes successful at kth retransmission.

4.2.1.2. Analysis of CQI fabrication.

We first discuss the *consecutive fabrication*, where a fabricator consecutively inflates its CQI. Next, we study a smarter fabricator, who tries to minimize the OLLA's penalty by sporadically inflating CQIs. Note that OLLA requires some time of accumulating multiple ACK/NACK signals to conjecture the "right" CQI.

Impact of consecutive CQI inflation. We compare CQI inflation (*BLER 30*, *MERA*, and *MERAQ*) with no CQI inflation denoted as Normal *BLER 10*, where Fig. 4(a) and (b) show PoF and penalty counter, both for the selfish user, respectively. We observe that PoF is less than one, implying that the fabricator does not obtain selfish gain just with consecutive CQI inflation. This is because with the consecutive CQI inflation, the frame errors (thus NACK fabrications) tend to increase. Then OLLA detects an abnormal situation, and penalizes the fabricator by reducing the actual CQI value to meet the target BLER 10%. This is verified by Fig. 4(b), where the penalty counter is about 4 on average, largely exceeding 0. Thus, OLLA mostly detects the "wrong" CQIs reports. Fig. 4(c) depicts the scheduled time slots of a selfish and normal users, it showing that under

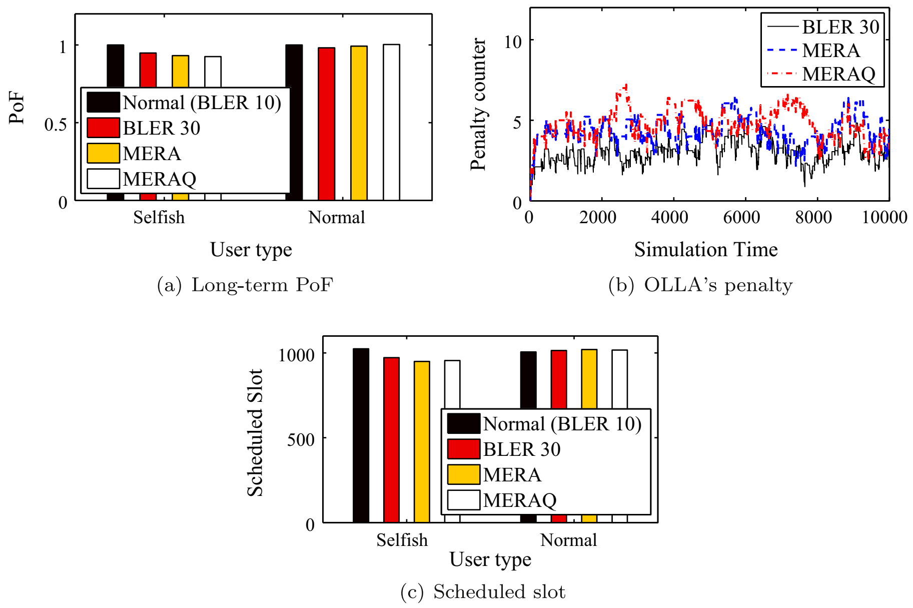


Fig. 4. Impact of consecutive CQI fabrication for selfish objective. (a) PoF of selfish user and normal users. PoFs are less than 1 implying that there is no fabrication gain by CQI fabrication. (b) OLLA's penalty for selfish user mostly exceeds 2 (i.e., more than two level CQI reductions). (c) Scheduled time slots. Normal user's scheduling opportunities is not significantly affected by the selfish user's CQI fabrication.

OLLA the selfish strategies do not negatively impact the scheduling opportunities of the normal user.

Impact of sporadic CQI inflation. A fabricator may be able to alleviate OLLA's penalty by sporadically inflating CQIs. To study its feasibility, we employ a simple, sporadic CQI fabrications that periodically switch the phases between inflation and non-inflation, where we test switching periods 10, 20, 50, 100, 200 time slots. We see that PoF is less than one for all cases, as shown in Fig. 5(a), implying that it is hard to obtain the fabrication gain even with the sporadic CQI fabrication. The reason is that it takes very long time to reduce and re-initialize the penalty counter of OLLA after being increased by CQI inflation, because OLLA responses to NACKs much more aggressively than it does to ACKs. As an example, in our scenario, two NACKs caused by CQI inflation makes OLLA decrease one CQI level. However, to increase one CQI level, 20 ACKs from the fabricator are needed (see Fig. 2). Since there are 10 users in a single cell, the time required to send 20 ACKs by the fabricator roughly need 20×10. Until sending the required ACKs for increasing OLLA's CQI, the fabricator keeps being penalized by OLLA. Thus, when we plot the fabricator's penalty counter curve, it repeatedly becomes positive and negative. Fig. 5(b) depicts the average durations of being positive and negative in the penalty counter, showing how OLLA aggressively penalizes the fabricator even for sporadic CQI inflation.

4.2.2. Malicious objective

Single User Fabrication. We now examine the impact of CQI fabrication for a malicious objective, where the fabricator's major goal lies in blocking normal users' service for a duration of consecutive time slots. A natural way of blocking normal users' service is that a malicious user estimates other users' scheduling index (i.e., $r_u(t)/R_u(t)$),[4] and selects its fabricated CQI so as to maximize the index and thus be scheduled by the BS. However, the malicious impact of such a single user fabrication turns out to be marginal, as shown in Fig. 6(a) for the case when the number of attackers is one, since (i) the scheduling priority of the malicious user quickly decreases due to the upper limit of CQI and the increase of $R_u(t)$ value via inflated CQIs (see the update rule of $R_u(t)$ of PFS in (2)), and (ii) OLLA even penalizes the fabricator's CQI boost-up. Note that in Fig. 6(a), we simulated 50 users with the similar environment to that in [12], and varied the number of colluding users from 1 to 5.

Group Fabrication. To maximize the malicious effect, a group of users may cooperate and carry out a group-wise, cooperative fabrication. The authors in [12] studied so-called *Delta CQI attack* and shows its significant malicious effect. To briefly explain how Delta CQI attack works, (i) every malicious user who is supposed to collude computes the increment of CQI value, denoted by $\delta_u(t)$, needed in order to be assigned a higher priority than other normal users by comparing to its previous CQI (i.e., $\delta_u(t) = CQI_u(t) - CQI_u(t-1)$), and (ii) the malicious user with the smallest $\delta_u(t)$ in the group reports its CQI to the BS while other malicious users report randomly low CQIs to the BS. Delta CQI attack tries to slow the increment of calculated CQI values for upcoming slots, thus the consecutive time slots that the malicious group can get from scheduler increases.

However, as Fig. 6(a) shows, our finding is that the malicious effect of Delta CQI attack is still small, which significantly differs from the analysis in [12]. We observe that the

[4] To do this, we can use a method of overhearing other users' feedback, as also mentioned in [12].

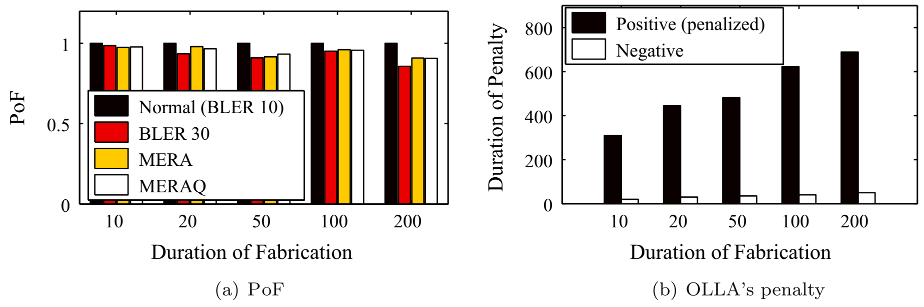


Fig. 5. Impact of sporadic CQI fabrication for selfish objective. (a) PoF value with three sporadic CQI fabrications, *BLER-30, MERA*, and *MERAQ*, where the tested fabrication durations are 10, 20, 50, 100, 200 time slots. In all cases, no inflation gain is observed. (b) Average durations of being positive and negative in the penalty counter, showing the aggressiveness of OLLA's penalization even for sporadic inflations. OLLA responses to NACKs much more aggressively than it does to ACKs.

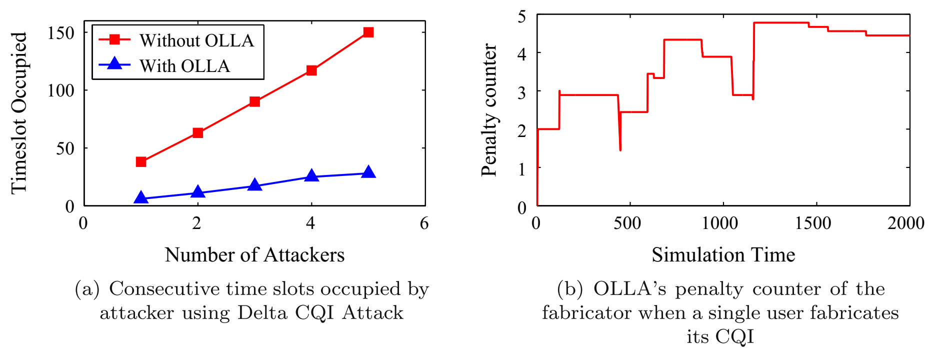


Fig. 6. OLLA mitigates the malicious effects by CQI fabrication. (a) Under OLLA, the consecutive time slots occupied by the malicious user significantly decreases compared to that without OLLA. (b) Due to the fabricated CQI, OLLA's penalty exceeds 0 to reduce the inflated CQI.

monopolized consecutive time slots are 6 vs. 38 for a single fabricator and 150 vs. 30 for five cooperative fabricators. Here, we find that OLLA again plays a key role of reducing the fabricator's CQI values as a penalty, and significantly mitigates the malicious attack via CQI inflation, whereas the work in [12] did not consider OLLA.

4.3. ACK fabrication

In ACK fabrication, our major interest lies in whether OLLA's CQI inflation due to fake ACKs (thus higher sending rate by BS, but more frame errors, due to CQI's boost-up) helps with increasing a fabricator's throughput or not. We employ simple, random ACK fabrications of sending a fake ACK with probabilities 0.4, 0.5, or 0.6, whenever it is supposed to send NACK for each frame error. As shown in the left side of Fig. 7(a), there exists almost no fabrication gain for all three cases, and PoF decreases with increasing ACK fabrication probabilities. To understand what happens, see Fig. 7(b), where, as expected, fake ACKs results in negative penalty counters, whose absolute value roughly quantifies how much CQI is boosted up (i.e., CQI is inflated due to fake ACKs). This increases not only the instantaneous data rate but also aggregate transmission rates at BS, as shown in Fig. 7(d) and (c). Note that this twofold increase does not guarantee that the fabricator's throughput is increased by ACK fabrication because of the missing frames caused by fake ACKs the fabricator sent. Here, the key reason of no fabrication gain is more missing frames due to fake ACKs (see the right side of Fig. 7(a). BLER increases with increasing ACK fabrication probabilities due to too much inflated CQI and it makes many missing frames). However, ACK fabrication may be a good method to minimize OLLA's penalty and thus may subsidiarily help the CQI inflation to obtain some selfish gain, which will be discussed in Section 5.

4.4. NACK fabrication

Explosive NACK fabrication has been used as a way of blocking normal users' service by exploiting the temporal priority change that the retransmitted frames get higher priority over normal frames [17]. However, OLLA is completely ignored in [17], and the conclusion on PFS vulnerability to NACK fabrication that PFS provides a large portion of scheduling chances to the fabricator was drawn. Motivated by their arguably wrong conclusion, to defend against malicious NACK fabricators, the authors in [17] proposed a modified PFS with a different $R_u(t)$ update rule, especially for each retransmission. In this section, we provide the analysis using simulation results that if OLLA is considered, significantly different PFS' behaviors against NACK fabricators are observed, from those in [17], and thus a different defense mechanism

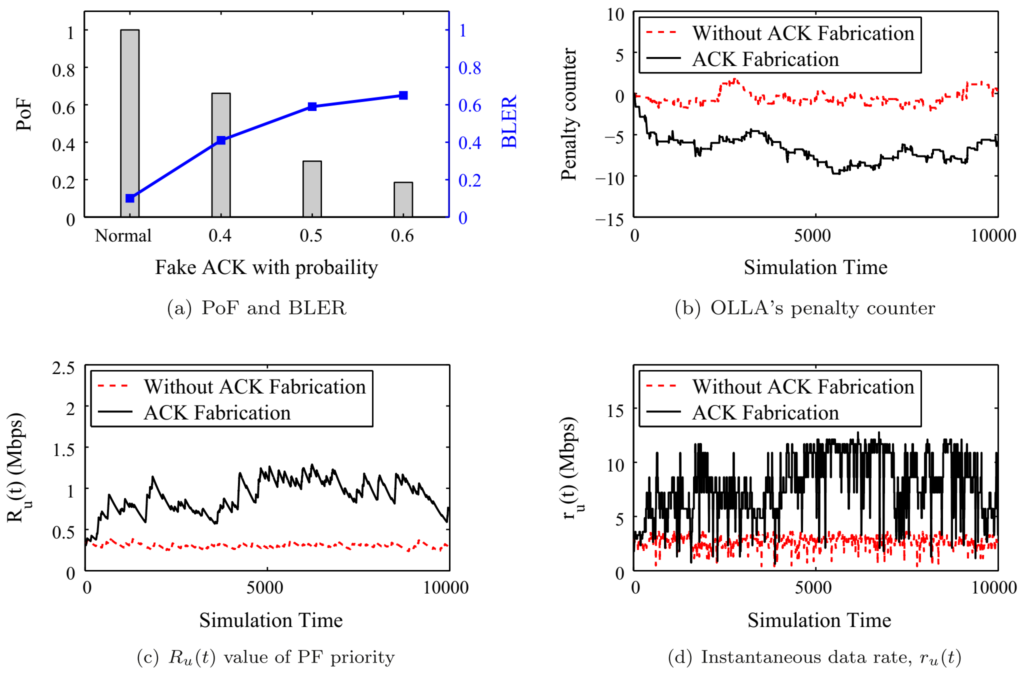


Fig. 7. Impact of ACK fabrication. (a) PoF and BLER for random ACK fabrications with fabrication probabilities 0.4, 0.5, and 0.6, where we observe no fabrication gains for all cases. The more aggressive the fabricator uses fake ACKs, the more missing frames occurs with higher BLER. (b) Penalty counter of the ACK fabricator, where negative counter values are traced by ACK fabrications (i.e., CQI inflation). (c) The averaged service rate, $R_u(t)$. (d) The instantaneous data rate, $r_u(t)$, where we observe that both rates increase due to ACK fabrications, but it does not guarantee that the fabricator can increase it's throughput via ACK fabrication.

can be applied. To that end, we examine PFS's behaviors for NACK fabricators, by dividing into consecutive and sporadic NACK fabrications.

Impact of consecutive NACK fabrication. The key message here is that fabricated NACKs are in malicious effect only after a large number of consecutive NACKs (e.g., thousands of slots) are reported to the BS. Consecutive fake NACKs lead to OLLA's penalization. This results in reduction of the instantaneous rate $r_u(t)$, but also growth of inter-scheduling times for the fabricator, say u, as shown in Fig. 8(a) over time slots $[0, 4000]$. Thus, OLLA seems to be capable of completely nullifying NACK fabrication. However, as fabrication continues (see Fig. 8(a), when time slot exceeds 4000), the following interesting things occur, so that normal users' service can be significantly starved, and thus high malicious effects, as explained next: After a number of time slots (4000 in our case), the fabricator's CQI cannot be penalized further because of the existence of the minimum CQI, and OLLA's penalty operation stops being in effect (see Fig. 8(b)). Then, the PF priority factor, $\frac{r_u(t)}{R_u(t)}$, increases again (see Fig. 8(c)), because $r_u(t)$ decreases just up to the minimum value, but $R_u(t)$ keeps decreasing. This incurs the starvation of normal users, who will be served very infrequently. Fig. 8(d) depicts that the malicious user can block normal users by consecutive NACK fabrication after 4000 time slots.

To understand more intuitively, we show a diagram in Fig. 9. For a fabricator sending fake NACKs to BS, OLLA at BS tries to penalize the user and reduces the instantaneous data rate, $r_u(t)$. This leads to the reduction of averaged service rate, $R_u(t)$ because of PFS's $R_u(t)$ update rules (see (2) in Section 2.3). Since both $r_u(t)$ and $R_u(t)$ values decrease, but the reduction speed of $r_u(t)$ reduction highly exceeds that of $R_u(t)$, the fabricator should have low priority in scheduling, which, in turn, reduces the chances to send fake NACKs. This happens until the fabricator's CQI is minimized, after which service blocking of normal users becomes possible, because once $r_u(t)$ meets the minimum value and thus no further OLLA penalization is possible, $R_u(t)$ keeps decreasing due to fake NACKs. This increases the PFS priority $\frac{r_u(t)}{R_u(t)}$, as depicted in Fig. 10, where the red arrows show the recursive sequence caused by NACK fabrication.

Impact of sporadic NACK fabrication. A fabricator may sporadically fabricate NACK because a BS can detect a malicious user who sends too many consecutive NACKs and exclude the user from scheduling for fairness among users. Similarly to the sporadic CQI inflation, we study NACK fabrications that periodically switch the phases between sending fake NACKs with periods 100, 1000, 3000 time slots (we also plot the results of "Cont." that continuously fabricates in the figures). Our main interest lies in the (maximum or average) consecutive time-slot duration during which normal users are starved. Fig. 11(a) plots when the fabricator is scheduled for various NACK fabrication periods. This shows that it is hard for the fabricator to significantly block normal users' service with less than 4000 time-slot period. Fig. 11(a) supports such a claim by showing that sporadically malicious user can be given only about 10 consecutive scheduling slots whereas

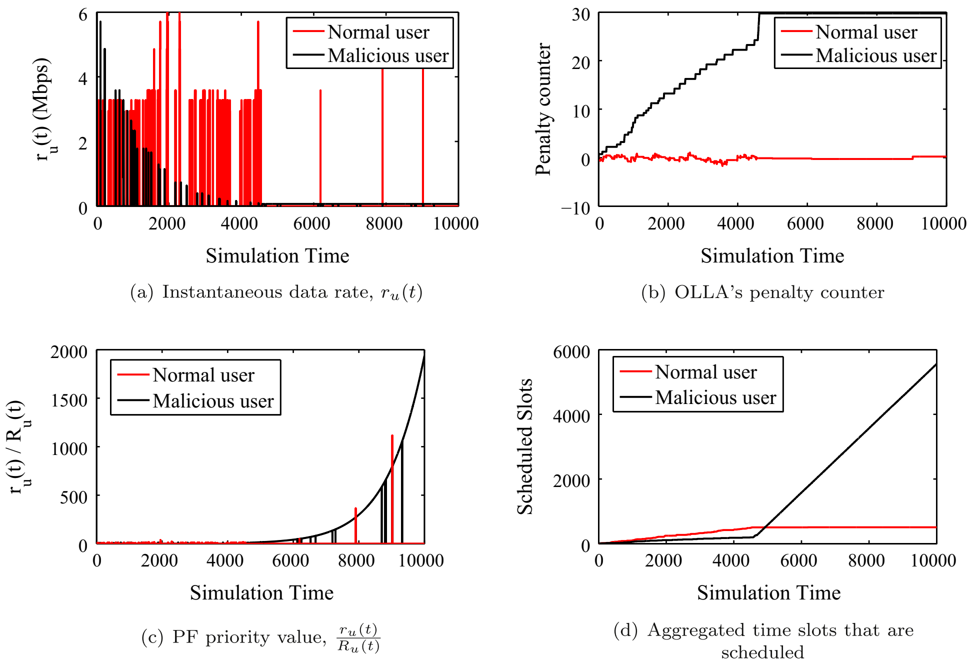


Fig. 8. Impact of consecutive NACK fabrication. OLLA mitigates the effects of NACK fabrications for a certain duration of time slots (as long as thousands of slots). However, after that duration, OLLA cannot mitigate NACK fabrications.

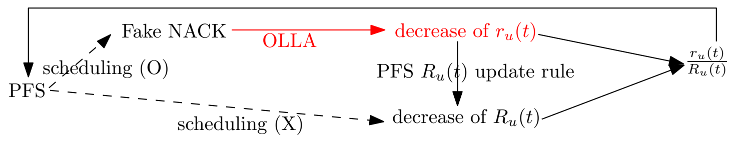


Fig. 9. Diagram that explains how NACK fabrication with OLLA behaves.

consecutive fake NACK incurs about 6000 time slot of blocking normal users.[5]

5. Analysis of joint fabrication schemes: OLLA-aware ones

Through the studies of single fabrications in earlier sections, most single CQI fabrication fails to obtain fabrication gains for either selfish or malicious purpose and single ACK fabrication also fails to provide selfish gains, mainly because OLLA appropriately adjusts its CQI against fake feedbacks in conjunction with the practical static BLER and HARQ-related retransmissions. In addition, for NACK fabrication, it turns out that OLLA can mitigate the impact of malicious users for a certain duration. We now consider smarter fabrications that jointly use a multiple of feedback fabrications; In particular, we consider two approaches: (i) fake NACK + minimum CQI for malicious users and (ii) CQI inflation + fake ACK for selfish users. Both fabrications are conceived with objective of mainly weakening OLLA's penalization by reporting minimum CQI or fake ACK.

5.1. Joint fabrication schemes

Malicious objective (fake NACK + minimum CQI). A malicious user strategically sends minimum CQI (i.e., CQI deflation), when it fabricates NACKs. This removes OLLA's penalization, because no further decrease of CQI becomes possible. This speeds up the process of making the malicious user have low averaged service rate (i.e., $R_u(t)$), which, in turn, leads to a burst of scheduling opportunities to the malicious user, and thus starving other normal users.

Selfish objective (CQI inflation + fake ACK). A selfish user sporadically uses fake ACKs to hinder OLLA from significantly lowering down CQIs, where without fake ACKs, CQI inflation is not in effect due to OLLA, as discussed in the earlier section. We naturally have the following three types of joint fabrication algorithms based on different CQI inflation schemes.

[5] One time slot is 2 ms in 3G HSDPA release 5 device [5], so 20 ms (10 slots) may be tolerable in most applications [12]. However, 12,000 ms (6000 slots) can significantly disrupt normal users, e.g., TCP throughput reduction or bad quality in VoIP.

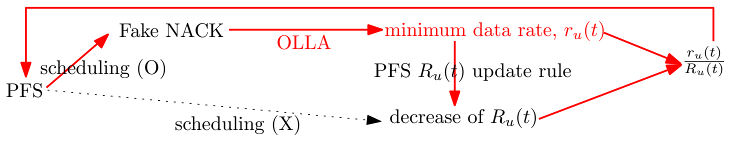


Fig. 10. Diagram that explains how NACK fabrication with OLLA behaves, once CQI reaches the minimum.

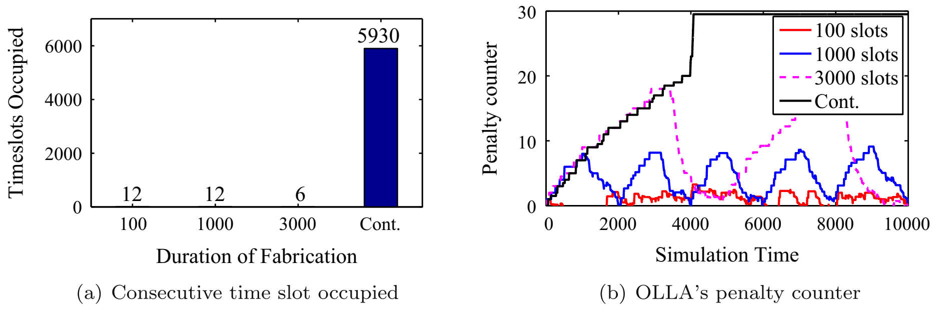


Fig. 11. Impact of sporadic NACK fabrication. (a) Scheduled slots for various NACK fabrication schemes. "Cont." means the consecutive NACK fabrication method. Since the OLLA can mitigate the NACK fabrication during initial 4000 time slots, the NACK fabrication during less than 4000 time slots occupy few time slots due to OLLA. (b) Penalty counter for a malicious user. OLLA mitigates the NACK fabrication except that the malicious uses "Cont." which is the consecutive NACK fabrication. For "Cont." case, the penalty counter cannot increase any more because it reaches the maximum after 4000 time slots.

J1. (*BLER-x* + fake ACK): This method uniformly fabricates ACK with $(x-10)$% replacing x% NACK frames to meet target 10% NACK ratio. For example, for BLER-30, a NACK is replaced by a ACK with probability 2/3.[6]

J2. (*MERA* + fake ACK): In this method, we divide the entire time slots into phases, where a phase consists of w time-slots. At the ith phase, we first compute the aggregate portion of NACKs up-to $i-1$ phase, say x% and NACKs are fabricated, such that BS is misled into believing only less than 10% frame errors occur. For example, when $x = 15\%$, at phase i, each NACK is replaced by a fake ACK to meet the target error rates, 10%.

J3. (*MERAQ* + fake ACK): This is similar to J2 except that MERAQ is used instead of MERA.

5.2. Fake NACK + minimum CQI

Recall that from Section 4.4 just fake NACK-based scheme requires long time (longer than 4000 slots in our simulation scenario) to increase PFS priority value. However, we observe that fake NACK + minimum CQI can quickly increase PFS priority value, as demonstrated in Fig. 12(a). This fast increase of PFS priority in the joint fabrication results in monopolizing BS's scheduling chances (thus starving other normal users, see Fig. 12(b)). The reason is explained by the following inter-plays: (i) Starting with the minimum CQI leads to the immediate decrease of the average service rate ($R_u(t)$), (ii) Smaller average service rate let the scheduling priority increase, which in turn increases the number of fake NACKs, and finally (iii) more fake NACKs speed up the decrease of the average service rate. Since the instantaneous rate $r_u(t)$ is strategically set to be the minimum, a malicious user is capable of obtaining a large share of BS scheduling chances. Note that with only fake NACK-based scheme, only fake NACK is a driver that lowers down $r_u(t)$, where fake NACK is generated only when the user is scheduled. This requires long time to have high scheduling priority in the pure fake-NACK based fabrication. This case of fabrication effects are well explained by the diagram in Fig. 10.

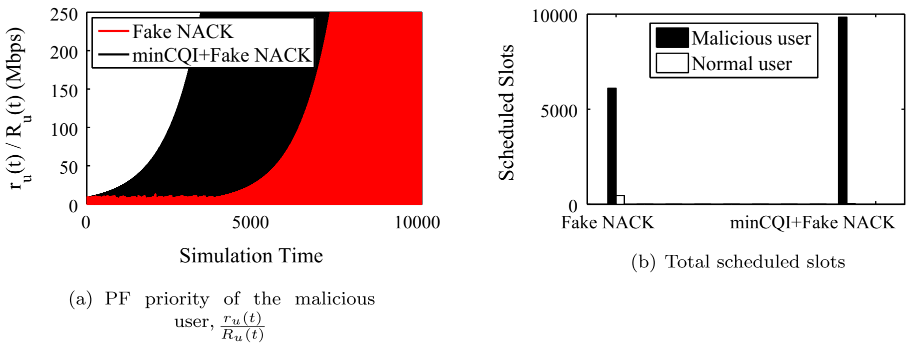
5.3. Fake ACK + CQI inflation

We now consider a joint CQI inflation and fake ACK based fabrication for selfish users. We analyze the selfish gain from the perspective of two layers: link and transport. There may exist the case where link-level throughput increases due to fabrication, yet transport-level one does not due to the existence of loss-sensitive transport protocols. In all simulations here, we combine the link-level simulator in Section 4.1 with TCP and UDP implementations in NS-2 using Enhanced UMTS Radio Access Network Extensions code [31].

Link-level. Recall that just CQI inflation methods (both sporadic and consecutive) can not achieve selfish gain because of OLLA's penalization. As shown in Fig. 13(a), J1, J2, and J3, have about 1.15 PoF value on average while only CQI inflation have less than 1 PoF value.[7] Thus, the fabricator who jointly mix fake ACKs with CQI inflation can get larger throughput than that of the normal users. In particular, see the J3 achieves about 1.2 PoF which is the largest among other fabrications. This is because MERAQ chooses the link

[6] This replacement is made only for the first transmission of a frame, not its retransmissions. This is also applied to J2 and J3.

[7] The selfish gains of J1, J2, and J3 come from that current MCS selection, BLER 10%, is not optimal for the channel conditions and these selfish strategies can bypass OLLA's penalty operation whereas BLER 30, MERA, and MERAQ cannot achieve selfish gain because of OLLA's penalty operation. Recall that most of the communication systems in practice uses BLER 10% for implementation simplicity.



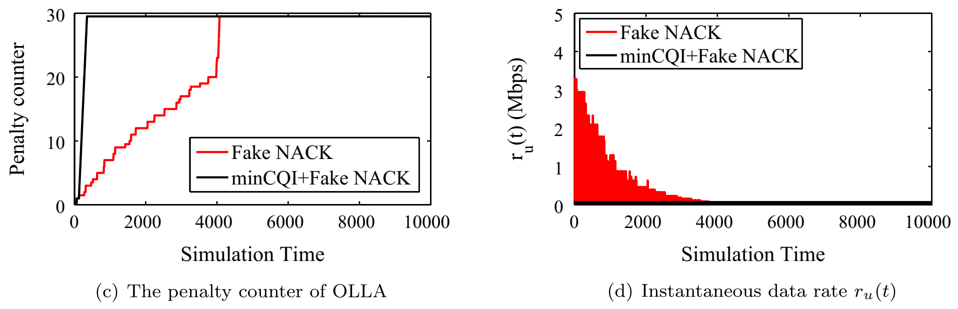


Fig. 12. Impact of fake NACK + minimum CQI. (a) PFS priority, $\frac{r_u(t)}{R_u(t)}$. (b) Total time share of joint fabrication and NACK fabrication. (c) Penalty counter of OLLA for the fabricator. (d) Instantaneous data rate, $r_u(t)$. All these plots show that starting with minimum CQI when fabricating NACKs increases the malicious user's scheduling priority in a short time, significantly blocking other users' service.

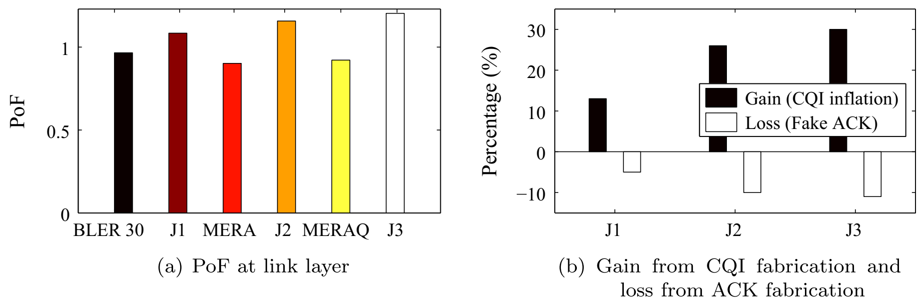


Fig. 13. Link-level impact of Fake ACK + CQI Inflation. (a) Positive fabrication gains are observed due to fake ACKs which mitigates OLLA's penalization. (b) The gain from inflated CQI exceeds the frame loss from fake ACK.

rate more aggressively than other fabrications by jointly considering HARQ retransmission while fake ACK helps to mitigate OLLA's penalty. Fig. 13(b) depicts the portions of the throughput gain by CQI inflation and the throughput loss by ACK fabrication for all schemes. The gain is the ratio of the average data rate of per-transmission with CQI inflation to that without fabrications denoted as percentage %. The loss is the ratio of the average missing data rates of per-transmission with ACK fabrication to that without fabrications. In our simulation scenario, the CQI inflation gain exceeds loss by fake ACKs because fake ACK mitigates OLLA's penalty operation so that the fabricator can fully exploit the gain from CQI inflation without CQI reduction. As an example, observe that J3 achieves about 30% of data rate increase by CQI inflation while about 10% of loss by ACK fabrication.

Transport-level. We now analyze the goodput of two representative transport-layer protocols. With the fake ACK based joint fabrication, transport-layer protocols are not strongly protected by MAC-level reliability, and thus a loss sensitive protocol such as TCP is highly likely to experience throughput degradation. Fig. 14(a) shows the PoFs of the selfish user who uses the J3 strategy for UDP and TCP, where in UDP, link-layer selfish gain is transferred to the transport layer without much degradation. However, in TCP, the throughput decreases due to the fabrication. As shown in Fig. 14(b), with fake ACKs, TCP experiences more congestion events (including a few retransmission timeouts due to the burst loss) than a normal TCP. Since most current Internet traffic uses TCP and the portion is around 90% [39] of the total traffic, it may

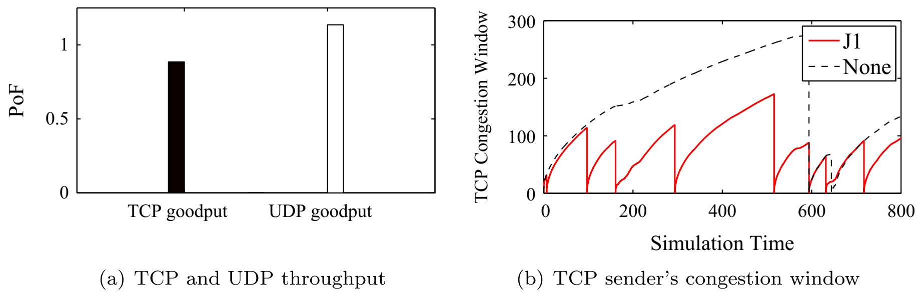


Fig. 14. Transport-level impact of Fake ACK + CQI Inflation. (a) Long-term transport-level PoF. (b) Congestion window of the selfish user's TCP. TCP's goodput decreases because fake ACK incurs the decrease of transmission rates (due to congestion avoidance and retransmission timeouts generated by packet losses).

seem difficult for a selfish user to increase its throughput by this fake ACK based joint fabrication.

6. Countermeasures

6.1. Malicious NACK fabrications

To defend against NACK fabricators based on our finding that malicious attack can work only after a large volume of NACK reports, we propose a mechanism that BS maintains the ACK/NACK history of window size w for each user, and calculates the percentage of NACK among the previous ACK/NACK history. For example, for $w = 30$, 15 NACKs and 15 ACKs for a user, its NACK percentage is 50%. If the NACK percentage exceeds some threshold, called *detection threshold*, then BS excludes the user from the service for some duration, which we call *exclusion duration*. BS temporarily removes the potentially malicious user from the service during the exclusion duration, and after that, checks again its NACK percentage to re-open the service to the user, only when the user does not report too many NACKs.

In the above algorithm as a countermeasure against NACK fabrications, there are three important parameters: (i) detection threshold, (ii) history size w, and (iii) exclusion duration. First, for detection threshold, our choice is 50% because, with the condition that w is not too small, the expected frame error of a normal user generally satisfies the target BLER (typically 10%). Second, for history size w, the w value is related to the false positive percentage which is the probability that BS misjudges a normal user, who temporarily reports consecutive NACKs due to shadowing from obstacles, as a malicious user. If w is small (i.e., small number of history), then the false positive percentage becomes large and if w is large (i.e., large number of history), then the false positive percentage is small. Our additional simulation results tell us that $w = 10$ is enough to have almost zero false positive percentage, shown in the Fig. 15 (see the simulation scenario in Section 4.1). Finally, if the exclusion duration is too long, the normal user who suffers from temporal shadowing effects can be removed from the service for a long period. In addition, from the fact that BS quickly can judge whether a user is malicious or not with small number of history (e.g., $w = 10$), the small exclusion duration is enough to control the malicious user who aggressively reports consecutive NACKs. In our scenario, we set the duration as 200 (time slots). Since there are 10 active users and PFS should serve each user with almost equal time shares, the malicious user will lose 20 (=200/10) scheduling opportunities. As shown in Fig. 16, NACK fabrication can be defended by the proposed countermeasure. Especially, as shown in the Fig. 16 (a), the malicious user cannot increase the PFS priority of user $u, \frac{r_u(t)}{R_u(t)}$ to monopolize scheduler by NACK fabrication due to that BS blocks the increase of PFS priority by periodically removing the malicious user from the service.

Note that the authors of [17] proposed a different defense mechanism that modified PFS with $R_u(t)$ update rule, named *Initial Effective Average*, which we do not think

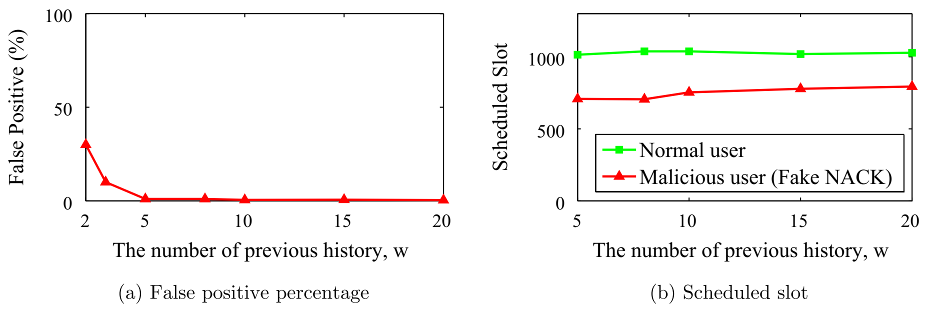


Fig. 15. Impact of ACK/NACK history size, w. (a) False positive percentage (b) Scheduled slot. The small history size (e.g., $w > 10$) is enough to differentiate between a malicious user and a normal user with nearly zero false positive rate. Under the proposed countermeasure, the malicious user gets less time share than a normal user even though the malicious user reports consecutive NACKs to grab the scheduler.

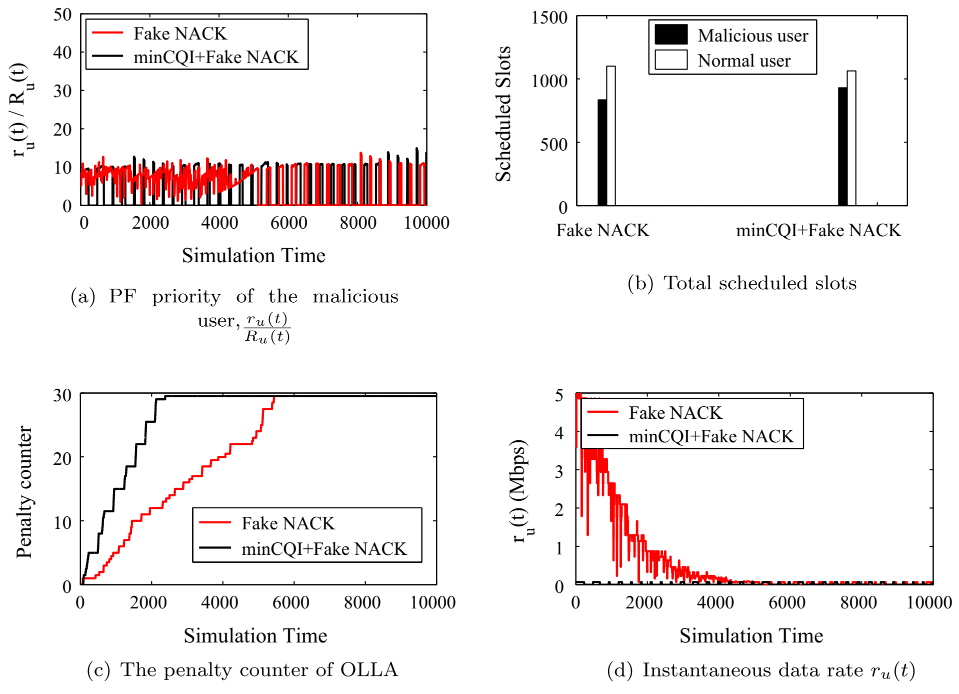


Fig. 16. Impact of countermeasure on fake NACK/minimum CQI + fake NACK. (a) PFS priority, $\frac{r_u(t)}{R_u(t)}$. (b) Total time share of joint fabrication and NACK fabrication. (c) Penalty counter of OLLA for the fabricator. (d) Instantaneous data rate, $r_u(t)$. The malicious user cannot get more time shares than a normal user by increasing the PFS priority of user $u, \frac{r_u(t)}{R_u(t)}$ to monopolize scheduler via NACK fabrication (+ minimum CQI). This is because that BS blocks the increase of PFS priority by periodically removing the malicious user from the service. The malicious user who fabricates only NACKs has less time shares than those who jointly use minimum CQI due to OLLA's penalty.

is necessary. In their update rule, the following are considered; (i) BSs update $R_u(t)$ even for the retransmitted frames so that the NACK fabricator cannot abuse retransmission to hold the scheduler, and (ii) they use the expected rate, $r_u^e(t)$, which is a data rate with considering the success probability of transmission (i.e., $r_u^e(t) = r_u(t)g_u(t)$ where $g_u(t)$ is the probability of successful transmission), instead of $r_u(t)$ when BSs update $R_u(t)$ for fairness. They showed that the Initial Effective Average rule is immune to the NACK fabrication, again on the basis of their "wrong" analysis that PFS becomes quickly vulnerable to starting from a small number of consecutive NACK feedbacks.

6.2. Joint fabrications

Countermeasure of fake NACK + minimum CQI. The countermeasure of NACK fabrication, where BS removes the malicious user reporting too many NACKs, can be applied here. However, malicious users who fabricates only NACKs may have less time shares than those who jointly use minimum CQI due to OLLA's penalty, see Fig. 16.

Countermeasure of fake ACK + CQI inflation. We can come up with a method which inspects whether the reported ACK is faked or not, based on the patterns only available to BS and randomly generated by BS. Once

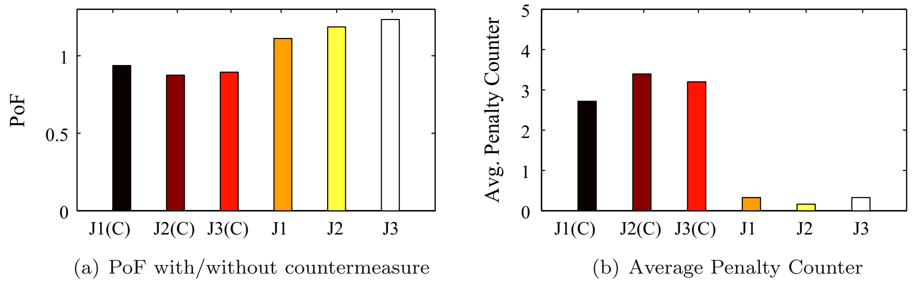


Fig. 17. Impact of countermeasure on fake ACK + CQI inflation. (a) PoF (for the fabricator). (b) Average penalty counter of OLLA (for the fabricator). J1, J2, and J3 are joint fabrication methods in Section 5. J1 (C), J2 (C), and J3 (C) are the same joint fabrication with countermeasure. Since BS with the proposed countermeasure can know whether a ACK is faked or not, OLLA cannot be weakened by fake ACK any more, thus, OLLA increases the penalty counter to penalize the fabricator who reports inflated CQI, so it is difficult for the fabricator to get larger PoF than 1 (i.e., it is difficult to increase the fabricator's throughput via fake ACK + CQI inflation).

receiving a frame, a user decodes the frame and returns the pattern (i.e., a small number of symbols) with ACK to the BS. The fabricator often cannot decode the frame successfully, and thus is unable to figure out the pattern, failing to return a "correct" ACK to the BS. Under this defense mechanism, BS easily detects the user who pretends to successfully receive frames by reporting the fake ACK, and BS can regard the fake ACKs as NACKs. Simulation results in Fig. 17(a) and (b) show that such countermeasure with OLLA efficiency defends against the fake ACK + CQI inflation.

7. Conclusion and discussion

In this paper, we jointly considered AMC, HARQ, OLLA, and PFS as a whole and revisited the impact of CQI, ACK, and NACK feedback fabrications under practical system configuration. Our simulation-based analysis for single fabrication leads that *(a)* selfish gains or malicious gains via CQI fabrication are hard to achieve in practice, which is a contrast to prior work, *(b)* selfish gains via ACK fabrication exploiting OLLA's behavior to boost-up its CQI is hard to achieve, and *(c)* the fabricated NACKs are in malicious effect only after a large number of consecutive NACKs (e.g., thousands of slots) are reported to the BS. In order to solve the vulnerability against NACK fabrication, we propose a simple mechanism that BSs regard the user who sends NACKs above some threshold as malicious and exclude the user from the service, as opposed to a complex mechanism based on the modification of the aggregate throughput in prior work. We also study smarter fabrications that jointly use multi-feedbacks. In particular, we study two joint fabrications: Fake NACK + minimum CQI for malicious objective and CQI inflation + fake ACK for selfish objective, through which we show that a smarter joint fabrication for selfish objective achieves the increase of its own average throughput at the cost of increasing upper-layer packet losses (thus no TCP-level selfish gain) and show that NACK fabrication with minimum CQI fabrication arrives faster the state where the other normal users are starved than only NACKs fabrication does. Our studies in this paper provide more practical answers on the question of how PFS is vulnerable to selfish and malicious feedback fabrications, where many findings lead to ones which significantly differ from those in related work.

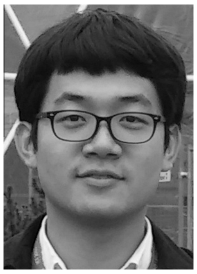
Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2013R1A2A2A01067633). This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Plannig (NRF-2014M3C4A7030648).

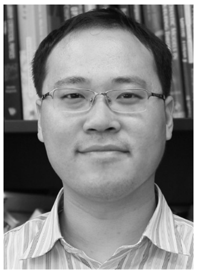
References

[1] A. Jalali, R. Padovani, R. Pankaj, Data throughput of cdma-hdr a high efficiency-high data rate personal communication wireless system, in: Proceedings of IEEE VTC Spring, 2000.
[2] J. Yang, Z. Yifan, W. Ying, Z. Ping, Average rate updating mechanism in proportional fair scheduler for hdr, in: Proceedings of IEEE GLOBECOM, 2004.
[3] H. Kushner, P. Whiting, Convergence of proportional-fair sharing algorithms under general conditions, IEEE Trans. Wireless Commun. 3 (4) (2004) 1250–1259.
[4] M. Andrews, Instability of the proportional fair scheduling algorithm for hdr, IEEE Trans. Wireless Commun. 3 (5) (2004) 1422–1426.
[5] A.T. Harri Holma, HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobile Communications, Wiley, 2006.
[6] P.J.B. Eduardo Esteves, M.I. Gurelli, Link adaptation techniques for high-speed packet data in third generation, in: Proceedings of EW, 2000.
[7] N. Bhushan, C. Lott, P. Black, R. Attar, Y.-C. Jou, M. Fan, D. Ghosh, J. Au, Cdma2000 1 times;ev-do revision a: a physical layer and mac layer overview, IEEE Commun. Mag. 44 (2) (2006) 37–49.
[8] D. Astely, E. Dahlman, A. Furuskar, Y. Jading, M. Lindstrom, S. Parkvall, Lte: the evolution of mobile broadband, IEEE Commun. Mag. 47 (4) (2009) 44–51.
[9] A. Ghosh, D. Wolter, J. Andrews, R. Chen, Broadband wireless access with wimax/802.16: current performance benchmarks and future potential, IEEE Commun. Mag. 43 (2) (2005) 129–136.
[10] OsmocomBB Project. <http://bb.osmocom.org/trac/>.
[11] G. Bianchi, A. Di Stefano, C. Giaconia, L. Scalia, G. Terrazzino, I. Tinnirello, Experimental assessment of the backoff behavior of commercial ieee 802.11b network cards, in: Proceedings of IEEE INFOCOM, 2007.
[12] R. Racic, D. Ma, H. Chen, X. Liu, Exploiting opportunistic scheduling in cellular data networks, in: Proceedings of NDSS, 2008.
[13] R. Racic, D. Ma, H. Chen, X. Liu, Exploiting and defending opportunistic scheduling in cellular data networks, IEEE Trans. Mobile Comput. 9 (5) (2010) 609–620.
[14] D. Kim, Y.-C. Hu, A study on false channel condition reporting attacks in wireless networks, in: Proceedings of ICST Securecomm, 2010.
[15] D. Kim, B. Jung, H. Lee, D. Sung, H. Yoon, Optimal modulation and coding scheme selection in cellular networks with hybrid-arq error control, IEEE Trans. Wireless Commun. 7 (12) (2008) 5195–5201.
[16] H. Zheng, H. Viswanathan, Optimizing the arq performance in downlink packet data systems with scheduling, IEEE Trans. Wireless Commun. 4 (2) (2005) 495–506.
[17] U. Ben-Porat, A. Bremler-Barr, H. Levy, B. Plattner, On the vulnerability of the proportional fairness scheduler to retransmission attacks, in: Proceedings of IEEE INFOCOM, 2011.
[18] K. Pelechrinis, P. Krishnamurthy, C. Gkantsidis, Towards a trustworthy pf scheduler for cellular data networks, in: Proceedings of IEEE GLOBECOM, 2012.
[19] G. Caire, D. Tuninetti, The throughput of hybrid-arq protocols for the gaussian collision channel, IEEE Trans. Inf. Theory 47 (5) (2001) 1971–1988.
[20] J.-F. Cheng, Coding performance of hybrid arq schemes, IEEE Trans. Commun. 54 (6) (2006) 1017–1029.
[21] C. Lott, O. Milenkovic, E. Soljanin, Hybrid arq: theory, state of the art and future directions, in: Proceedings of Information Theory for Wireless Networks Workshop, 2007.
[22] D. Chase, A combined coding and modulation approach for communication over dispersive channels, IEEE Trans. Commun. 21 (3) (1973) 159–174.
[23] D. Paranchych, M. Yavuz, A method for outer loop rate control in high data rate wireless networks, in: Proceedings of IEEE VTC Fall, 2002.
[24] K. Pedersen, F. Frederiksen, T. Kolding, T. Lootsma, P. Mogensen, Performance of high-speed downlink packet access in coexistence with dedicated channels, IEEE Trans. Veh. Technol. 56 (3) (2007) 1262–1271.
[25] M. Nakamura, Y. Awad, S. Vadgama, Adaptive control of link adaptation for high speed downlink packet access (hsdpa) in w-cdma, in: Proceedings of WPMC, 2002.
[26] A.M. Cavalcante, E.B. Souza, J.J. Bazzo, M.J. Souza, L. Kuru, J. Moilanen, On the system-level analysis of outer loop link adaptation for ieee 802.16e systems, J. Commun. Comput. (2012).
[27] K. Aho, O. Alanen, J. Kaikkonen, Cqi reporting imperfections and their consequences in lte networks, in: Proceedings of ICN, 2011.
[28] A. Pokhariyal, T. Kolding, P. Mogensen, Performance of downlink frequency domain packet scheduling for the utran long term evolution, in: Proceedings of IEEE PIMRC, 2006.
[29] K. Pedersen, G. Monghal, I. Kovacs, T. Kolding, A. Pokhariyal, F. Frederiksen, P. Mogensen, Frequency domain scheduling for ofdma with limited and noisy channel feedback, in: Proceedings of IEEE VTC Fall, 2007.

[30] 3GPP. 3G specification: 3GPP TR 36.942 v8.1.0, Tech. Rep., 2008.
[31] Enhanced UMTS Radio Access Network Extensions for ns-2, <http://eurane.ti-wmc.nl/>.
[32] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Trans. Autom. Control 37 (12) (1992) 1936–1948.
[33] V. Kavitha, A. Eitan, R. El-Azouzi, R. Sundaresan, Opportunistic scheduling in cellular systems in the presence of non-cooperative mobiles, in: Proceedings of IEEE CDC, 2009.
[34] V. Kavitha, A. Eitan, R. El-Azouzi, R. Sundaresan, Fair scheduling in cellular systems in the presence of noncooperative mobiles, in: Proceedings of IEEE INFOCOM, 2010.
[35] 3GPP, 3G specification: 3GPP TS 25.214 v11.4.0, 2012.
[36] I.T. Union, Guidelines for evaluation of radio transmission technologies for IMT-2000. Tech. Rep., ITU Std. Rec. ITU-R M.1225, 1997.
[37] K. Freudenthaler, A. Springer, J. Wehinger, Novel sinr-to-cqi mapping maximizing the throughput in hsdpa, in: Proceedings of IEEE WCNC, 2007.
[38] K. Ko, D. Lee, M. Lee, H.-S. Lee, A novel sir to channel-quality indicator (cqi) mapping method for hsdpa system, in: Proceedings of IEEE VTC Fall, 2006.
[39] K. chan Lan, J. Heidemann, A measurement study of correlation of Internet flow characteristics, Comput. Networks 50 (1) (2006) 46–62.



Hanjin Park received the B.S degree in Computer Science from the Yonsei University, Seoul, Korea in Feb. 2007. He is currently working toward the Ph.D. degree at the Department of Computer Science, KAIST. His research interests are in mobile ad hoc networks, delay tolerant networks, and wireless cellular network.



Yung Yi received his B.S. and the M.S. in the School of Computer Science and Engineering from Seoul National University, South Korea in 1997 and 1999, respectively, and his Ph.D. in the Department of Electrical and Computer Engineering at the University of Texas at Austin, USA in 2006. From 2006 to 2008, he was a post-doctoral research associate in the Department of Electrical Engineering at Princeton University. Now, he is an associate professor at the Department of Electrical Engineering at KAIST, South Korea. His current research interests include the design and analysis of computer networking and wireless communication systems, especially congestion control, scheduling, and interference management, with applications in wireless ad hoc networks, broadband access networks, economic aspects of communication networks (aka network economics), and green networking systems. He was the recipient of two best paper awards at IEEE SECON 2013 and ACM Mobihoc 2013. He is now an associate editor of IEEE/ACM Transactions on Networking, Journal of Communication Networks, and Elsevier Computer Communications Journal.

Yongdae Kim is a professor in the Department of Electrical Engineering at KAIST. He received Ph.D. degree from the computer science department at the University of Southern California under the guidance of Gene Tsudik. He received his MS and BS degrees in Mathematics from Yonsei University in 1993 and 1991. Between 2002 and 2012, he was an associate/assistant professor in the Department of Computer Science and Engineering at the University of Minnesota - Twin Cities. Before joining U of Minnesota, he worked as a research staff for two years in Sconce Group in UC Irvine. Before coming to the US, he worked 6 years in ETRI for securing Korean cyberinfrastructure. He received NSF career award on storage security and McKnight Land-Grant Professorship Award from University of Minnesota in 2005. Currently, He is serving as a steering committee member of NDSS (Network and Distributed System Security Symposium). His current research interests include security issues for various systems such as social networks, cellular networks, P2P systems, medical devices, storage systems, mobile/ad hoc/sensor/cellular networks, and anonymous communication systems.