A Study on False Channel Condition Reporting Attacks in Wireless Networks*

Dongho Kim and Yih-Chun Hu

Electrical and Computer Engineering, University of Illinois at Urbana-Champaign
{dkim99,yihchun}@illinois.edu

Abstract. Wireless networking protocols are increasingly being designed to exploit a user's measured channel condition; we call such protocols *channel-aware*. Each user reports its measured channel condition to a manager of wireless resources and a channel-aware protocol uses these reports to determine how resources are allocated to users. In a channel-aware protocol, each user's reported channel condition affects the performance of every other user. A possible attack against channel-aware protocols is *false feedback* of channel condition. The deployment of channel-aware protocols increases the risks posed by false feedback. In this paper, we study the potential impact of an attacker that falsely reports its channel condition and propose a defense mechanism to securely estimate channel condition. We analyze our mechanism and evaluate the system performance deploying our mechanism through simulation. Our evaluation shows that our mechanism effectively thwarts channel condition misreporting attack.

Keywords: Wireless Network, Opportunistic Scheduler, Cooperative Relay.

1 Introduction

Many protocols in modern wireless networks treat a link's channel condition information as a protocol input parameter; we call such protocols *channel-aware*. Examples include opportunistic schedulers [1, 2], cooperative relaying network architectures [3, 4], and efficient ad hoc network routing metrics [5, 6]. Even though each different application exploits the channel-condition information in different ways, the main goal of a channel-aware protocol is to enhance system throughput by selecting a user or a path with good channel condition in a given time instance.

Most work on channel-aware protocols has mainly focused on how channel condition information can be used to more efficiently utilize wireless resources. An implicit assumption of most past study is that each user correctly reports channel condition information. However, this assumption can induce a security vulnerability since channel condition can be asymmetric [7]; specifically, due to

* This material is based upon work partially supported by USARO under Contract No. W-911-NF-0710287 and the NSF under Grant No. CNS-0953600.

S. Jajodia and J. Zhou (Eds.): SecureComm 2010, LNICST 50, pp. 107–124, 2010.

possible channel condition asymmetry, channel condition to a user *can only be measured and reported by that user*. An attacker that misreports its measured channel condition might allow the attacker to steal another users' service opportunities, for example in a setting where a centralized scheduler schedules each user based on its channel condition. In another setting, a user chooses a next-hop forwarder based on the relayer's channel condition, in which case an attacker can misreport its channel condition to generate a sinkhole [8] to lure packets to itself possibly for the purpose of dropping those packets.

In this paper, we reveal the possible effects of false channel condition reporting in various channel-aware network protocols and propose a defense mechanism that provides secure channel condition estimation. Our contributions are:

- We propose a secure channel condition estimation algorithm that is generally applicable to any channel-aware protocol.
- We analyze our algorithm in terms of performance and security, and we perform a simulation study to verify our performance analysis.
- We analyze the effect of misreported channel condition on reference systems with opportunistic schedulers and cooperative relaying protocols. We also show through simulation that our defense mechanism thwarts the attack effect on those systems.

The false channel condition reporting attack that we introduce in this paper is difficult to identify by existing mechanisms, since our attack is mostly protocol compliant; an attacker need to modify only the channel-condition measurement mechanism. Our attack can thus be performed using modified user equipment legitimately registered to a network.

To the best of our knowledge, we are the first to study the false channel condition reporting attack in a variety of network settings. Racic et al. [9] consider attacks based on false feedback to the PF scheduler. In their work, as in our work, PF effectively resists false feedback, so their attack primarily works by exploiting the handover process rather than the channel-aware nature of PF scheduler. They propose a secure handover algorithm that is orthogonal to our approach of secure channel condition estimation.

The rest of our paper is organized as follows. In Section 2, we introduce the concept of our attack. Then, we develop a defense mechanism called *secure channel condition estimation* against the false reporting attack in Section 3. We evaluate our algorithm through analysis and simulation in Section 4. In Section 5, we briefly review related work. Section 6 concludes this paper.

2 Attack Overview

Threat Model. Our study assumes that a network protocol exploits the channel condition information reported by each user and each user reports to enhance network performance. In this setting, a user can falsely report its channel condition. There are two different types of false reports: *underclaiming* (reporting a channel condition that is worse than that actually measured by the user) and

overclaiming (reporting a channel condition that is better than that actually measured by the user). The effectiveness of a false channel condition reporting attack depends on the way the attacked protocol uses the reported channel condition, and an attacker's ability to exploit the protocol. We use the term 'channel condition' to refer to all aspects impacting a node's ability to receive a packet.

Attack Purpose. Generally, an attacker's goal in a network is to greedily raise its own bandwidth share or to maliciously downgrade other users' bandwidth share without regard to its own bandwidth share. For these purposes, an underclaiming action is not desirable since underclaiming merely forfeits an attacker's service opportunity. Hence, we focus only on overclaiming actions in this paper. An overclaiming receiver may lose its throughput since the overclaiming receiver may induce a higher order (more aggressive) modulation, possibly resulting in excessive loss. As a result, this paper focuses on attackers that are malicious rather than selfish. We demonstrate through simulation the attack's effect on specific systems in Section 4.3.

Attack Feasibility. An attacker can easily implement false channel condition reporting attack by modifying only a subcomponent that reports channel condition. This subcomponent of user equipment can be implemented in hardware or software. One recent trend of user equipment implementation is to move increasing amount of functionality into software in order to improve adaptability [10, 11, 12]. The increasing software control of user equipment makes false channel condition reporting attack an increasingly practical attack.

3 Defense

In this section, we discuss possible solutions for the false channel feedback attack introduced in Section 2. We argue that to fundamentally defend against attacks that involve false channel condition reports, we need a scheme to securely estimate channel condition. Then, we develop our secure channel condition estimation algorithm.

3.1 Solution Spectrum

To defend against an attack that misreports the channel condition, there are possible approaches. One possible approach is anomaly detection. Anomaly detection is a tool that monitors each user's performance to identify attackers. A response mechanism then disconnects the attacker from the network. A second possible approach is to devise a fair scheduler to provide fair share of a network bandwidth while exploiting channel-aware property. A third possible approach is to measure throughput of a node and compare the measured throughput and the theoretically calculated throughput based on reported channel condition.

Even though these approaches can mitigate the effectiveness of the attack, they have fundamental drawbacks. Anomaly detection mechanisms are subject to detection errors, which could result in incorrect termination of a normal user's

service or failure to detect an attacker. When a fair scheduler is used to reduce the effect of the attack, we can frustrate the original goal of channel-aware protocol which is to use resources most efficiently. A scheduler considering fairness will substantially reduce the efficiency when compared to the original protocol, since fairness requires allocation of resources to less-capable channels. We will see the throughput difference between most efficient scheduler and fair scheduler in Section 4.3. In a setting where a sender chooses a relayer with good channel condition, a receiver can calculate a theoretical throughput based on a relayer's reported channel condition and compare the theoretical throughput and actually received throughput. However, this method assumes that a receiver is honest. A receiver can be an attacker not acknowledging that an honest relayer does not forward its packets.

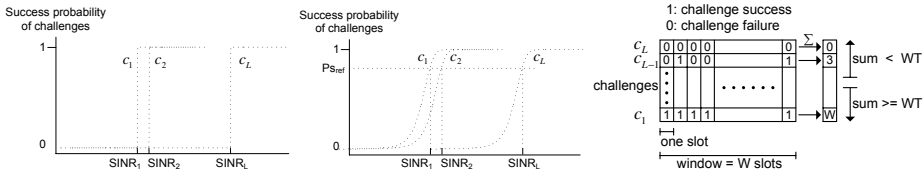
To more effectively prevent the false channel condition reporting attack, we need a mechanism that does not impede the efficiency of channel-aware protocols even under the false reporting attack. We observe that the false reporting attacks are possible because we allow a non-trustable entity to report the channel condition. Our basic approach is to replace non-trustable-entity-driven channel-condition reporting with trustable-entity-driven channel-condition estimation. For example, in a cellular network, base station is a trustable entity and users are non-trustable entities. In this paper, we do not develop whole specific protocols for such networks; rather, we develop a generic algorithm that can be integrated into any channel-aware protocol. We leave protocol integration and design as future work.

3.2 Scope of Our Algorithm

There are two cases to consider an channel condition misreporting attack. The first case is that a trusted entity gets the report of a node's channel condition to the trusted entity. The second case is that a trusted entity can get the report of a node's channel condition to the other node from an untrusted node. The latter case can happen in the deployment of an efficient ad hoc network routing metrics [5,6]. In the first case, the trusted entity can securely examine the channel condition using our algorithm. However, for the latter case, it is difficult for a trusted entity to identify an attacker since the trusted entity may not trust the reports from an untrusted node. In this paper, we focus on the first case as an initial step toward a complete defense against an attack.

3.3 Secure Channel Condition Estimation

In this section, we present our secure channel estimation scheme to prevent an attacker from overclaiming. We do not consider underclaiming, as explained in Section 2, because an attacker gains no benefit from underclaiming, and because an attacker can always reduce its actual channel condition, for example by modifying his antenna. The purpose of this paper is not to propose a whole system but to describe how our mechanism defends against an attacker that overclaims the condition of a single link. We start by presenting the intuition of our approach.



(a) Ideal Challenge Configuration (b) General Challenge Configuration (c) Channel Estimation

Fig. 1. Secure Channel Estimation

Intuition. For convenience of presentation, we call the trustable entity a "base station" and the non-trustable entity a "user". The base station's goal is to securely and accurately estimate each user's channel condition. We first present our solution to a simplified problem in which a base station wants to know whether or not a user experiences channel condition at least as good as some specified SINR. To solve this simplified problem, the base station sends a *challenge* to a user. This challenge is a packet that can be correctly decoded with high probability only when the channel condition exceeds some specified SINR. The challenge includes a value known only to the base station. Upon receiving the challenge, a user returns the value in that challenge to the base station, which can then compare the received value to the transmitted value. The base station considers the channel condition to exceed the specified SINR if and only if the received value is correct. This challenge mechanism prevents a user with poorer channel condition than the specified SINR from correctly decoding the challenge packet. Our channel condition estimation scheme extends this single challenge scheme to multiple challenges in order to more finely estimate the channel condition.

System Model. We consider a network cell consisting of a base station and N users served by the base station. $\mathcal{N} = \{1, 2, \ldots, N\}$ denotes the set of all users in the system. The base station estimates channel conditions of each user in each time slot using L challenges. A time interval $[dt - d, dt), t \in \mathbb{Z}$ is called time slot t where d is the duration of a time slot. At each time slot t, the base station uses our channel condition estimation to determine a user's channel condition as an element in a set $\mathcal{E} = \{E_1, E_2, \ldots, E_{L+1}\}$ with cardinality $L + 1$. Each element $E_i \in \mathcal{E}$ represents an SINR range of $\text{SINR}_{i-1} \leq \text{SINR} < \text{SINR}_i$, where $\text{SINR}_0 = -\infty$ and $\text{SINR}_{L+1} = \infty$.

Construction of Challenges. In our scheme, the base station sends challenges to users so that users cannot overclaim their channel condition. To prevent the overclaiming attack, a challenge must have the following properties: unpredictability of the value included in a challenge and a well-designed success probability curve of the challenge. If a user receiving a challenge is able to guess the challenge value, the user can return the correct value even without successfully decoding the challenge. To make the challenge value unpredictable, we use a pseudorandom number generator.

To make a challenge that can be successfully decoded only by users with channel condition above a specified SINR, the success probability curve of a challenge must be appropriately designed. The ideal success probability curve would have zero success probability for channel condition worse than a specified SINR and zero error probability for channel condition better than that specified SINR as shown in Fig. 1(a). The dotted lines represent the success probability of reception of challenges according to SINR. With these ideal challenges, the successful reception of a challenge c_i and the failure of the reception of c_{i+1} implies that a given channel condition is $\text{SINR}_i \leq \text{SINR} < \text{SINR}_{i+1}$. We could then estimate the channel condition as E_{i+1}. These ideal challenges enable us to easily and accurately estimate the channel condition with only a single transmission. However, ideal challenges require infinitely large challenges. Our scheme considers non-ideal challenges, as shown in Fig. 1(b). For each challenge c_i, a node with channel condition as the threshold SINR_i for that challenge will successfully decode the challenge with probability $Ps_{ref(i)}$. Even though the shapes of the success probabilities of each challenge look same in Fig. 1(b), our scheme does not require the shape of each success probability to be the same. We discuss the choice of $Ps_{ref(i)}$ for the optimal performance in Section 4.1.

An immediate method to construct multiple challenges having appropriate success probability is to use different modulation and coding techniques for each challenge. However, from a practical point of view, a particular system may not provide various modulation and coding options. In such cases, we need a method to construct challenges with the limited number of modulation and coding options available. In order to not interrupt the flow of presentation, we explain such methods in more detail in Section 3.5.

Transmission of Challenges. The base station periodically broadcasts a set of challenges to users. The period is one parameter of our scheme. One extreme is to send a set of challenges in a single time slot, which allows rapid channel condition estimation and can respond to rapid variations in channel condition. However, sending so many challenges results in significant overhead. In an environment where the channel condition is slowly changing, we can reduce the frequency with which a base station sends challenges.

Estimation. After the base station transmits a challenge to a user, the user returns the challenge value to the base station to prove that the channel to the user is good enough to receive the corresponding challenge. When the base station receives the value from the user, the base station checks that the value is identical to the one that it sent. Then, the base station stores the result of this check. We denote a check result for challenge c_i at time slot t by $F_i(t)$.

$$F_i(t) = \begin{cases} 0 \text{ if challenge } c_i \text{ failed} \\ 1 \text{ if challenge } c_i \text{ succeeded} \end{cases}$$

With ideal challenges, only a single set of check results is enough to estimate channel condition. Since our scheme uses non-ideal challenges, we need multiple sets of check results to reduce the error in the estimated channel condition.

We call the set used for estimating channel condition a window, and we denote the window size as W. Intuitively, a larger window size results in more accurate estimated channel condition but slower adaptation. In Section 4.1, we theoretically analyze the impact of window size on the performance of our algorithm. When a base station finishes collecting a window of check results $F_i(t-W+1), \ldots, F_i(t), \forall i \in \{1, \ldots, L\}$ at time slot t, the base station sums the check results for each challenge $c_i, \forall i \in \{1, 2, \ldots, L\}$ as follows.

$$S_i(t) = \sum_{j=0}^{W-1} F_i(t-j) \qquad \forall i \in \{1, 2, \ldots, L\}$$

Based on the values of $S_i(t)$, the base station estimates channel condition using a decision function D. In other words, the base station decides which element in the set $\mathcal{E} = \{E_1, E_2, \ldots, E_{L+1}\}$ most accurately characterizes corresponding user's channel condition. We denote the estimated channel condition at time slot t by $E_c(t)$.

$$E_c(t) = D(S_1(t), S_2(t), \ldots, S_L(t))$$

We use a simple threshold-based comparison for our decision function D. Fig. 1(c) shows the comparison procedure. We choose a threshold $T \in [0, 1]$. First, we see how any of the lowest rate challenges (c_1s) are successfully received by a user; it is likely that nearly all of these challenges are received by the user because it checks the lowest SINR range. When all c_1s are successfully received, $S_1(t) = W$. If $S_1(t) \leq WT$, we proceed to check $S_2(t)$. We repeat until we reach $S_i(t) < WT$. That is, we pick $i = \min j, \text{s.t.} S_j(t) < WT$. The base station then estimates the channel condition $E_c(t) = E_i$. For this threshold-based comparison, it is important to choose a proper threshold T. We analyze the impact of T on performance of our algorithm in Section 4.1.

3.4 Application of Our Secure Estimation Algorithm

There are two application types of our secure channel condition estimation algorithm. First, our algorithm can be used to detect and penalize an attacker by comparing reported channel condition to estimated channel condition. We do not pursue this approach further since it suffers from false detection like anomaly detection. Second, our algorithm can be used to select a node with good channel condition. The purpose of this approach is not to penalize an attacker but to provide a fair service to every node. After a node is chosen by our secure estimation algorithm, a sender node can determine modulation order by seeing reported channel condition to reduce loss probability. In this approach, an overclaiming attacker does not gain any benefit since the attacker will the same amount of service opportunity as other users (if all users experience same channel condition) and loss probability will be higher than other users due to higher modulation order.

3.5 Implementation of Multiple Challenges

As discussed in Section 3.3, we need a way to construct multiple challenges having different success probability curves using the limited number of given modulation and coding options. In this section, we introduce two methods to reshape the success probability of a challenge.

The first method is processing gain [13] which improves SINR by transmitting the same signal multiple times; when these copies add up, the signal energy increases by more than the noise power, thus increasing the SINR and shifting the success probability curve higher. To explain the concept of processing gain more formally, we rely on communication theory. We assume that a signal $s(t)$ is transmitted through an Additive White Gaussian Noise (AWGN) channel $n(t)$. The AWGN channel is a channel model which distribution is normal distribution. We assume that in our channel, mean is zero and variance is σ^2 ($\sim N(0, \sigma^2)$). The variance is considered to be noise power. SINR is calculated in symbol time (T) basis. When two identical signals are transmitted, the signal energy is $\int_0^T |2s(t)|^2\, dt = 4\int_0^T |s(t)|^2\, dt$. Hence, the energy of two signals is four times (6dB) higher than that of a single signal. The addition of two AWGN sources is considered to be the sum of two normal distributions ($N(0, \sigma^2) + N(0, \sigma^2) = N(0, 2\sigma^2)$). Hence, the noise power ($2\sigma^2$) of two signals is two times (3dB) higher than that (σ^2) of a single signal. Consequently, the ratio of signal energy to noise power of the sum of two signals is two. With the addition of two signals, we can shift a success probability curve of a challenge to left by 3dB. With the larger number of signal additions, we can shift the success probability curve further to left.

The second method is to add noise in a signal at the transmitter. By adding a noise to a signal, we can reduce the ratio of signal energy to noise power of a signal. Hence, we can shift a success probability curve a challenge to right.

4 Evaluation

In this section, we evaluate the performance and the security of our algorithm. First, we analyze the impact of algorithm parameters on the performance of our algorithm. This analysis can be used to guide our parameter choices. We then perform simulations and compare the result of our analysis to those of our simulation. Second, we integrate our algorithm into a network simulator and evaluate the effect of our algorithm on system performance. We show that our algorithm securely and effectively estimates channel condition through most of its parameter space. Third, we analyze the security of our algorithm. In this analysis, we show that an attacker cannot, by guessing the value of a challenge, cause the channel condition estimate to be higher than if the attacker decoded the challenge in the same way as a normal user. In other words, *regardless of the length of a challenge value*, an attacker and a normal user that experience equivalent channel conditions will receive equal channel estimates in expectation. This paper does not include an evaluation of the overhead that our algorithm imposes. We leave as future work an exploration of the trade-off between the accuracy of estimation and system overhead.

4.1 Performance Analysis

In this section, we analyze the effect of parameter choices on our channel condition estimation algorithm. Specifically, we derive average estimation error $E[|\text{CQI} - \widehat{\text{CQI}}|]$ based on algorithm parameters such as window size (W), threshold (T), the size of a challenge and $Ps_{ref(i)}$ of a challenge. CQI (Channel Quality Indicator) in the average estimation error equation represents an actual CQI-level. $\widehat{\text{CQI}}$ represents an estimated CQI-level.

Assumptions. Our analysis assumes that the channel condition does not change. To analyze variable channel condition, we need to enumerate all possible cases for channel conditions in multiple slots. This analysis requires excessive amounts of computing power. Hence, we use simulation to consider the effect of variable channel condition in Section 4.2. The equations in our analysis do not assume the same values of challenge size and $Ps_{ref(i)}$ for each challenge. However, allowing different values of challenge size and $Ps_{ref(i)}$ increases the parameter space substantially. So when we plot figures, we use the same challenge size and $Ps_{ref(i)}$ for all challenges.

Analysis. Given a target SINR which is mapped to a CQI, we calculate the probability distribution on the estimated CQI ($\widehat{\text{CQI}}$), and then we calculate average estimation error.

We start by assuming that we have functions $R_i(\text{SINR}, Ps_{ref(i)}), \forall i \in \{1, 2, \ldots, L\}$ representing the probability that a bit of a challenge c_i is successfully received given an SINR. This function depends on the modulation and coding method used for constructing challenges, and is well-understood in communication theory [14]; we later illustrate numerical results with a specific modulation and coding scheme. The probability P_{cs_i} that a challenge c_i is successfully received is calculated as

$$P_{cs_i} = R_i(\text{SINR}, Ps_{ref(i)})^{SC_i}$$

where SC_i is the length in bits of challenge c_i. The number of successful challenges in a window of size W for challenge c_i is binomially distributed with probability P_{cs_i}. Hence, the probability $P_{c_i}(n)$ of exactly n successful challenges can be expressed as

$$P_{c_i}(n) = \binom{W}{n} P_{cs_i}^n \left(1 - P_{cs_i}\right)^{W-n}$$

We can now calculate the probability $P_{ec}(i, \text{SINR})$ that CQI is estimated to be i given SINR. $\widehat{\text{CQI}} = i$ represents that E_{i+1} is chosen by our algorithm. Our algorithm estimates CQI by comparing the number of successful challenge receptions to the product of window size and threshold WT. Counting the number of successful challenge receptions from the lowest CQI-level, our algorithm determines $\widehat{\text{CQI}} = i$ when the number of successful challenge receptions for CQI-level i is less than WT. For CQI-level less than i, the number of successful challenge receptions is greater than or equal to WT. Hence, $P_{ec}(i, \text{SINR}), \forall i \in \{0, \ldots, L-1\}$ is calculated as

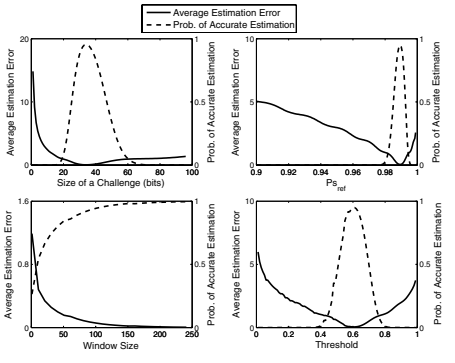


Fig. 2. Average estimation error and estimation accuracy for various parameters

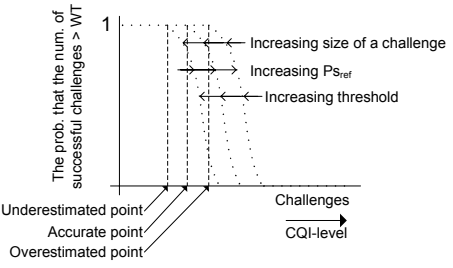


Fig. 3. Analyzing parameter design

$$P_{ec}(i,\mathrm{SINR}) = \prod_{j=1}^{i} \left(P_{c_j}(\lceil WT \rceil) + P_{c_j}(\lceil WT \rceil + 1) \cdots + P_{c_j}(W)\right) \times \left(1 - P_{c_{i+1}}(\lceil WT \rceil) - P_{c_{i+1}}(\lceil WT \rceil + 1) \cdots - P_{c_{i+1}}(W)\right)$$

For CQI-level L, we have a different form.

$$P_{ec}(L,\mathrm{SINR}) = \prod_{j=1}^{L} \left(P_{c_j}(\lceil WT \rceil) + P_{c_j}(\lceil WT \rceil + 1) + \cdots + P_{c_j}(W)\right)$$

With $P_{ec}(i,\mathrm{SINR})$, we can obtain the average estimation error as follows.

$$E[|\mathrm{CQI} - \widehat{\mathrm{CQI}}|] = \sum_{i=0}^{L} |\mathrm{CQI} - i|\, P_{ec}(i,\mathrm{SINR})$$

Using this analysis on average estimation error, we now want to properly set window size, threshold, the size of a challenge, and reference probability $Ps_{ref(i)}$ of a challenge so that the average estimation error is minimized. As discussed in our assumptions, we use the same values of challenge size and $Ps_{ref(i)}$ for different challenges for ease of performance comparison. To obtain specific numerical results, we use the same definition of CQI as in the 3GPP standard [15].

$$\mathrm{CQI} = \begin{cases} 0 & \mathrm{SINR} \leq \text{-16dB} \\ \lfloor \frac{\mathrm{SINR}}{1.02} + 16.62 \rfloor & \text{-16dB} < \mathrm{SINR} < 14\mathrm{dB} \\ 30 & \mathrm{SINR} \geq 14\mathrm{dB} \end{cases}$$

This CQI configuration is also used for following simulations. Since P_{ec} has a non-continuous function (ceil function), it is difficult to apply optimization theory. To search for optimal parameters in the discontinuous space, we used

Table 1. block size (bits) for channel condition

cqi	block	cqi	block	cqi	block	cqi	block	cqi	block
1	137	2	173	3	233	4	317	5	377
6	461	7	650	8	792	9	931	10	1262
11	1483	12	1742	13	2279	14	2583	15	3319
16	3565	17	4189	18	4664	19	5287	20	5887
21	6554	22	7168	23	7168	24	7168	25	7168
26	7168	27	7168	28	7168	29	7168	30	7168

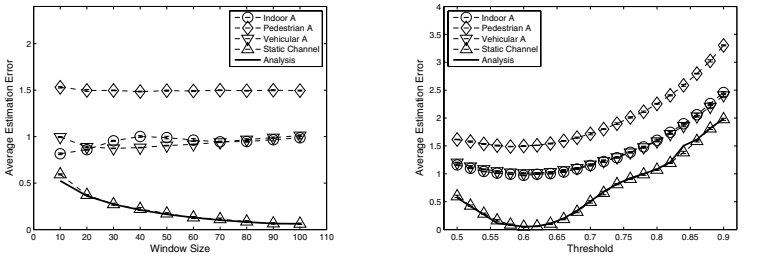
a hill-climbing approach [16]. First, we set initial values for each parameter intuitively. We then iteratively picked a parameter, optimized this parameter leaving all other parameters fixed, and repeated this process until we converged on a locally optimal parameter set. In the following results, we started with this parameter set and varied parameters one at a time to explore the impact of each parameter on system performance. Our calculation uses the reception probability for QPSK as $R_i(\text{SINR}, Ps_{ref(i)})$. We choose target SINR to allow for equal amounts of overestimation and underestimation in terms of CQI-level; in UMTS, this corresponds to a CQI level of 15 and an SINR of -1.19dB.

Fig. 2 shows our calculated average estimation error and the probability of accurate estimation. The results show an optimal point for each parameter: the size of a challenge, reference probability of a challenge, window size, and threshold. To demonstrate why the optimal points exist, we show the probability that the number of successful challenge transmission is greater than WT for each challenge in Fig. 3. As the size of a challenge increases, the probability curve slides towards the direction of underestimation. Increasing the threshold moves the probability curve in the same direction as the size of a challenge. With increasing reference probability, the probability curve moves towards the direction of overestimation. For window size, larger window size provides a better accuracy. This is intuitively obvious, since the large window size provides larger number of test samples for estimating channel condition.

4.2 Simulation

We performed a simulation study to verify our analysis and consider the effect of variable channel condition on the performance of our algorithm. We start with the case of a static channel condition.

Static Channel Condition. We implemented our algorithm in the NS-2 simulator [17] patched with EURANE [18], a UMTS system simulator. Our reference system is a UMTS system. To obtain specific numerical results, we use the same CQI configuration as we used to obtain numerical results for our analysis. Table 1 shows the transmission block sizes for each corresponding CQI [15]. As in the numerical results of our analysis, we consider the verification process for an SINR of -1.19dB which is a CQI of 15, and modulation using QPSK. We use the same optimal parameter selection as we used in the analysis in Section 4.1. We use the default UMTS time slot duration of 2ms, and our algorithm estimates the channel condition in each time slot. We vary window size and threshold,



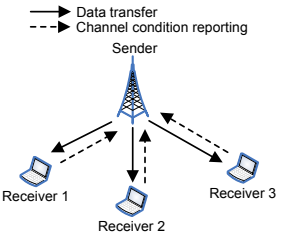
(a) Varying window size (b) Varying threshold

Fig. 4. Simulation results

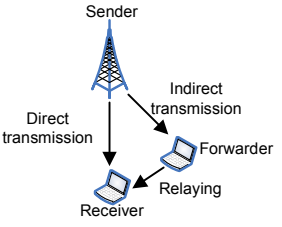
fixing $Ps_{ref(i)}$ and the length of each challenge value. We perform five runs for each value of window size and threshold.

For each estimation, we record the difference between actual CQI and estimated CQI (in absolute value). Fig. 4(a) and Fig. 4(b) show the average value of the differences, and validate the results of our analysis. As window size increases, the average estimation error decreases as expected. For all values of window size, the estimation error is below 1 CQI-level, and decreases to 0.05 CQI-levels as the window size grows to 100. Even larger windows would further reduce the error. However, our results show that our algorithm performs accurately with a reasonable window size.

Variable Channel Condition. Even though we can adjust parameters to optimize estimation accuracy in environments with static channel condition, the same parameter setting does not guarantee the same accuracy under a variable channel condition. We use a variable condition channel model to evaluate the effectiveness of our algorithm under a variable channel condition. We repeat the previous simulations, replacing the static channel condition with three UMTS channel models [19]: Indoor A with velocity 3km/h, Pedestrian A with velocity 15km/h and Vehicular A with velocity 120km/h. Fig. 4(a) shows the effect of window size on average estimation error. The average estimation error in variable channel conditions is greater than the error in a static channel condition, and the window size has significantly less impact on accuracy than in a static channel condition; this shows that the variability of the channel condition prevents our algorithm from achieving arbitrary precision by indefinitely increasing the window size. Nonetheless, in most cases, our algorithm's error is not greater than 1 CQI-level. Furthermore, both legitimate nodes and attacking nodes experience similar errors, further reducing the effectiveness of overclaiming. Fig. 4(b) shows the average estimation error for various values of threshold. Again, the estimation error in a static channel condition is less than the errors in variable channel conditions. However, our result shows that we can find a value of threshold that limits the estimation error less than 1.5 CQI-levels, and that the optimal parameters for static channel condition confies to be effective under variable channel conditions.

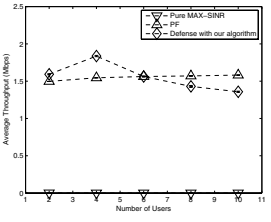


(a) Opportunistic Scheduler: Scheduler chooses a receiver in each time slot based on users' channel conditions.

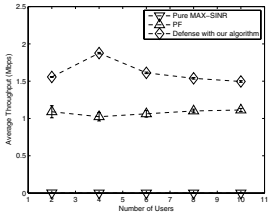


(b) Relaying Network: A sender forwards a packet to a relayer when the relayer has better channel condition than the receiver.

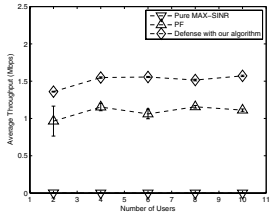
Fig. 5. Example Networks For System Performance Evaluation



(a) Indoor A



(b) Pedestrian A



(c) Vehicular A

Fig. 6. The effect of our algorithm on opportunistic scheduler

4.3 System Performance

So far, we have evaluated the performance of our secure channel estimation algorithm. Now, we evaluate the impact of our secure channel condition estimation algorithm on system performance. This evaluation provides an understanding on how much the estimation error of our algorithm affects the system performance. Our reference system is the system that we used for the previous simulation in Section 4.2. We implement opportunistic scheduler and cooperative relaying in this reference system. As we mentioned in Section 3.2, our algorithm works in the case where a trustable entity gets the report of channel condition of a node to the entity. Hence, we consider single-hop cooperative relaying network where each user reports its channel condition to base station. We do not consider an efficient routing metric [5,6] in ad hoc network where channel condition between intermediate nodes are reported to a source node.

Opportunistic Scheduler. Figure 5(a) shows how opportunistic scheduler [1,2] works in a wireless network. An opportunistic scheduler is a centralized resource scheduler that exploits the channel condition information of each user for efficient

resource management. One simple example of an opportunistic scheduler is an efficiency-oriented scheduler that allocates resources to only the user with the best channel condition in a time slot. We call this scheduler MAX-SINR. It is obvious that this scheduler achieves the maximum possible system throughput. However, this scheduler may give so few opportunities to a user with poor channel condition that it induces a fairness problem. The Proportional Fair (PF) scheduler [1] is a widely known scheduler that addresses the fairness problem. The PF scheduler collects channel condition information from each user at each time slot t. The PF scheduler uses channel condition feedback from each user to determine which user to serve by calculating metrics $R_i(t)/T_i(t)$ for each user, where $T_i(t)$ is user i's average throughput calculated as

$$T_i(t) = \begin{cases} (1 - 1/t_c)T_i(t-1) + 1/t_c R_i(t) & \text{if user } i \text{ is chosen} \\ (1 - 1/t_c)T_i(t-1) & \text{if user } i \text{ is not chosen} \end{cases}$$

and t_c represents the time constant of a low pass filter. In each time slot, the PF scheduler serves the user with the largest metric.

Our simulated network consists of one base station serving several users, half of which are attackers. The attackers choose a simple attack: overclaiming their channel condition to be the best possible condition. The base station reacts by choosing a high bit-rate modulation for each transmission to any attacker, which can induce a high error rate when the actual channel condition is poor. In EURANE's implementation, a node that is unable to receive a packet would not send back an ack to the base station, triggering an internal control mechanism in UMTS that stops any connection failing to acknowledge several contiguous transmissions. We modified the attacker to send an ack for every received packet, whether or not that packet was received without error. Our channel model for each user is the same variable channel models (Indoor A, Pedestrian A, and Vehicular A) that we used for performance analysis. We sourced 11 Mbps of CBR traffic to each user. We measure the throughput of normal users under three scheduling policies: PF, MAX-SINR without our algorithm and MAX-SINR with our algorithm. In MAX-SINR with our algorithm, a base station does not use user-reported CQI-level to pick which user has the best channel condition in a give time slot. Instead, the base station uses the CQI-level estimated by our algorithm. Figure 6 shows that MAX-SINR is vulnerable to overclaiming attack and PF prevents attackers from stealing normal users' throughput. However, our simulation results show that MAX-SINR with our algorithm can achieve higher throughput than PF scheduler in most cases. Occasionally PF outperforms MAX-SINR, because our algorithm occasionally overestimates the receiver's channel condition, in which case the base station may choose a modulation scheme that is too aggressive, resulting in packet loss.

Cooperative Relaying Network. In a mobile wireless network, mobile nodes can experience different channel conditions due to their different locations. Though a node experiences a channel condition too poor to receive packets from a source node, a third node may have a good channel condition to both the source and the intended destination. *Cooperative relaying* network architectures (e.g., [3,4,20,21])

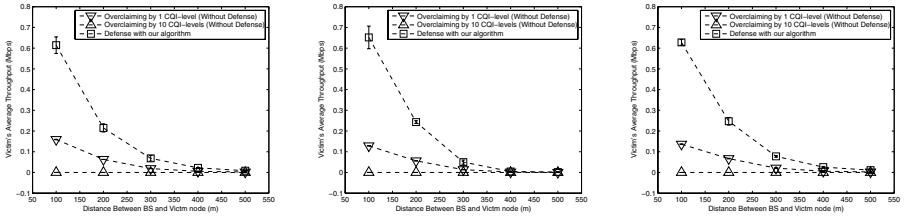
shown in Figure 5(b) help a node that has poor channel condition to route its packet through a node with a good channel condition, thus improving system throughput. In order to find such routes, a cooperative relaying protocol must distribute channel condition information for each candidate path, find the most appropriate relay path, and provide incentives to motivate nodes to forward packets for other nodes. Specifically, in UCAN [4], user equipment has two wireless adaptors, one High Data Rate (HDR) cellular interface and one IEEE 802.11 interface. The HDR interface is used for communication with a base station and the IEEE 802.11 interface is used for peer-to-peer communication with other user equipment in a network.

In our simulated network, the base station is the traffic source. The victim node chooses a relayer node if the relayer has a better channel condition. An attacking relayer node overclaims its channel condition to intercept packets to the victim node. As shown in Section 4.2, the channel model can affect the estimation error of our algorithm. Hence, we use the three channel models (Indoor A, Pedestrian A, and Vehicular A) that we used for the simulation of variable channel condition. These results will show us how the estimation error due to varying channel condition affects the system performance. As we vary the distance between the base station and the victim node, we measure the victim's throughput under the false channel condition reporting attack.

Figure 7 shows the measured results in the case of a single relayer for the victim node. Hence, the attacker node is the only relayer for the victim node. In Figure 8, there are two relayers and one relayer is an attacker node. We consider three different cases: overclaiming by 1, overclaiming by 2 and defense with our algorithm. For the cases of overclaiming by 1 and 2, we plot the results without deployment of our algorithm. For the case of defense with our algorithm, we deployed our algorithm to compare the cases with defense and without defense. When the victim node is close to the base station, the throughput of the case with defense is much greater than the throughput of the case without defense. As the victim node is farther from the base station, the throughput difference between defense case and non-defense case is reduced. It is because the degraded channel condition for the victim node far from the base station induces small capacity for the victim node. In the case of two relayers, we can see that due to the redundant relayer, the attack effect is reduced. Over three different channel models, we can see that the throughput results are similar to each other. With these results, we believe that the estimation error of our algorithm does not affect the system performance so much.

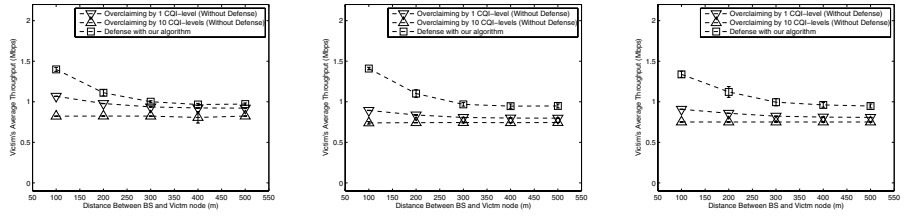
4.4 Security Analysis

The security of our scheme for securely estimating channel condition relies on the assumption that the attacker cannot predict the challenge values generated by a pseudo-random number generator. An attacker, then, has two strategies by which he can generate replies: either the attacker can guess the challenge value, or the attacker can attempt to decode the received challenge value as a normal user would. In this section, we will show that when the challenge values are chosen using a pseudo-random number generator, decoding is the dominating strategy of an attacker.



(a) Indoor A (b) Pedestrian A (c) Vehicular A

Fig. 7. The effect of our algorithm on a relaying network example (one relayer)



(a) Indoor A (b) Pedestrian A (c) Vehicular A

Fig. 8. The Effect of Our Algorithm on A Relaying Network Example (Two Relayers)

We assume that a data symbol experiences an Additive White Gaussian Noise (AWGN) channel, which is a typical model. The optimal (maximum-likelihood) decoder under AWGN takes the input signal and provides the data symbol most likely to correspond to that signal. An attacker that guesses ignores the input signal entirely, and as such, throws away any information contained in the input signal. Discarding this information could not improve the attacker's expected performance, because otherwise the optimal decoder would not be optimal. In other words, the attacker gains no advantage by guessing instead of decoding.

To illustrate, we consider BPSK coding with a received power level of 1 and AWGN power σ. In this environment, the sender sends +1 to send a 1-bit and -1 to send a 0-bit. The receiver receives the sender's value plus a random value drawn from $N(0, \sigma^2)$. The optimal decoder decodes a 1-bit if the received value is greater than 0 and a 0-bit otherwise, which has probability of success $Q(-\frac{1}{\sigma})$. Since $\sigma > 0$, $Q(-\frac{1}{\sigma}) > 0.5$. By simply guessing a bit, an attacker is successful with probability 0.5. The success probability of decoding is always greater than or equal to the success probability of guessing. Hence, if the challenge values are randomly generated, the optimal strategy is to use the optimal decoder. This result shows that an attacker cannot outperform a normal user.

5 Related Work

In this section, we review attacks related to our reference systems.

Attacks on Opportunistic Schedulers. Bali et al. [22] reveal a vulnerability in the PF scheduler that can be induced by a malicious traffic pattern. Bursty traffic enables a single flow to occupy several consecutive slots. They measure this attack's effect on real EV-DO network. The work by Racic et al. [9] on PF scheduler is the closest work to ours in the sense that they consider the effect of falsely reporting channel condition. They conclude that falsely reporting channel condition alone does not do harm other users very much in networks using a PF scheduler. They do find that falsely reporting combined with handover can occupy many consecutive time slots, thereby stealing other user's opportunity to be served. Unlike this work, we find cases where false reporting channel condition alone can significantly affect other user's performance in other network settings.

Attacks on Hybrid Networks. A hybrid network is one that implements cooperative relaying using two distinct data link technologies. Carbunar et al. [20] propose JANUS for defending against selfish or malicious behavior in establishing routes in hybrid networks. They consider the possibility of a rate inflation attack in which a node reports a higher bandwidth to base station than the node can provide. However, their attack overclaims the output rate of a link rather than the channel quality. In JANUS a base station sends request packets to nodes, and uses the fact that an overclaimed link will experience congestive losses. However, JANUS' request packets are not cryptographically secured, so the attacker can guess when it needs to send a response packet to hide the attack from the base station. Our approach differs from the JANUS' in that our algorithm uses cryptographic security to protect challenge messages. More fundamentally, because our verification is conducted at the physical layer, it allows for a more fine-grained verification of channel condition. Haas et al. [21] propose SUCAN, which defends against Byzantine behaviors in hybrid networks. However, they do not consider attacks that misreport channel condition.

6 Conclusion and Future Work

In this paper, we have studied the threat posed by attacks that falsely report their channel condition. Our false channel-feedback attack can arise in any channel-aware protocol where a user reports its own channel condition. To counter such attacks, we propose a secure channel condition estimation algorithm to prevent the overclaiming attack. Through analysis and simulations, we show that with proper parameters, we can prevent the overclaiming attack.

The protocol we describe requires that a trusted entity sends each challenge message, and we present two case studies, the opportunistic scheduler and the single-hop cooperative relaying, in which the trusted entity naturally arises within the environment. In a multi-hop channel-aware protocol, an intermediate hop may have no incentive to correctly estimate channel condition or to correctly relay another link's estimated channel condition. In this paper, we have focused on the single-hop channel estimation environment as an initial step towards defense against channel condition misreporting attack, and we leave secure multi-hop estimation and reporting to future work.

References

1. Jalali, A., Padovani, R., Pankaj, R.: Data throughput of cdma-hdr a high efficiency-high data rate personal communication wireless system. In: Proc. IEEE VTC, vol. 3, pp. 1854–1858 (May 2000)
2. Viswanath, P., Tse, D.N.C., Laroia, R.: Opportunistic beamforming using dumb antennas. IEEE Transactions on Information Theory 48(6), 1277–1294 (2002)
3. Sendonaris, A., Erkip, E., Aazhang, B.: Increasing uplink capacity via user cooperation diversity. In: Proceedings of IEEE International Symposium on Information Theory, p. 156 (August 1998)
4. Luo, H., Ramjee, R., Sinha, P., Li, L.E., Lu, S.: Ucan: a unified cellular and ad-hoc network architecture. In: ACM MobiCom, pp. 353–367. ACM, New York (2003)
5. De Couto, D.S.J., Aguayo, D., Bicket, J., Morris, R.: A high-throughput path metric for multi-hop wireless routing. In: ACM MobiCom, pp. 134–146. ACM, New York (2003)
6. Draves, R., Padhye, J., Zill, B.: Comparison of routing metrics for static multi-hop wireless networks. In: ACM SIGCOMM, pp. 133–144. ACM, New York (2004)
7. Jing, T., Wang, H.J., Hu, Y.C.: Preserving location privacy in wireless lans. In: Proc. ACM MOBISYS (June 2007)
8. Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: Attacks and countermeasures. In: First IEEE International Workshop on Sensor Network Protocols and Applications, pp. 113–127 (2002)
9. Racic, R., Ma, D., Chen, H., Liu, X.: Exploiting opportunistic scheduling in cellular data networks. In: NDSS (2008)
10. Odyssey 8500, `http://www.wavesat.com/pdf/OD-8500-IC-PB.pdf`
11. Airspan, `http://www.airspan.com/products_wimax.aspx`
12. Sdr, `http://en.wikipedia.org/wiki/Software-defined_radio`
13. Smith III, J.O.: Spectral Audio Signal Processing. In: Center for Computer Research in Music and Acoustics, CCRMA (2009)
14. Proakis, J.: Digital Communications, 4th edn., McGraw-Hill Science/Engineering/Math (August 2000)
15. Physical layer procedures (fdd), release 5. 3GPP TS25.214 V5.5.0 (June 2003), `http://www.3gpp.org/ftp/Specs/archive/25_series/25.214/25214-550.zip`
16. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education, London (2003), `http://portal.acm.org/citation.cfm?id=773294`
17. ns-2: Network simulator, `http://www.isi.edu/nsnam/ns/`
18. eurane: Enhanced umts radio access network extensions for ns-2, `http://eurane.ti-wmc.nl/eurane/`
19. Selection procedures for the choice of radio transmission technologies of the umts. ETSI TS UMTS 30.03 V3.2.0
20. Carbunar, B., Ioannis, I., Nita-Rotaru, C.: Janus: A framework for scalable and secure routing in hybrid wireless networks. IEEE Transactions on Dependable and Secure Computing (2008)
21. Haas, J.J., Hu, Y.C.: Secure unified cellular ad hoc network routing. In: IEEE Globecom (2009)
22. Bali, S., Machiraju, S., Zang, H., Frost, V.: A measurement study of scheduler-based attacks in 3G wireless networks. In: Uhlig, S., Papagiannaki, K., Bonaventure, O. (eds.) PAM 2007. LNCS, vol. 4427, pp. 105–114. Springer, Heidelberg (2007)