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

Dongho Kim and Yih-Chun Hu

**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 the 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. The deployment of channel-aware protocols increases the risks posed by false channel-condition feedback. In this paper, we study what happens in the presence of an attacker that falsely reports its channel condition. We perform case studies on channel-aware network protocols to understand how an attack can use false feedback and how much the attack can affect network performance. The results of the case studies show that we need a secure channel condition estimation algorithm to fundamentally defend against the channel-condition misreporting attack. We design such an algorithm and evaluate our algorithm through analysis and simulation. Our evaluation quantifies the effect of our algorithm on system performance as well as the security and the performance of our algorithm.

**Index Terms**—Network-level security and protection, wireless communication

## 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 cooperative relaying network architectures [1], [2], efficient ad hoc network routing metrics [3], [4] and opportunistic schedulers [5], [6]. While work on channel-aware protocols has mainly focused on how channel condition information can be used to more efficiently utilize wireless resources [1]–[6], security aspects of channel-aware protocols have only recently been studied [7]–[9]. These works on security of channel-aware protocols revealed new threats in specific network environments by simulation or measurement. However, understanding the effect of possible attacks across varied network environments is still an open area for study.

In particular, we consider the effect of a user equipment's reporting false channel condition. This issue is partially addressed in the work of Racic *et al.* [7] in a limited network setting. They consider a particular scheduler in a cellular network with handover process and propose a secure handover algorithm. In contrast, we reveal the possible effects of false channel condition reporting in various channel-aware network protocols and propose a primitive defense mechanism that provides secure channel condition estimation. Our contributions are:

- We analyze specific attack mechanisms and evaluate the effects of misreporting channel condition on various channel-aware wireless network protocols including cooperative relaying protocols, routing metrics in wireless ad-hoc network and opportunistic schedulers.
- We propose a secure channel condition estimation algorithm that can be used to construct a secure channel-aware protocol in single-hop settings.
- We analyze our algorithm in the respects of performance and security, and we perform a simulation study to understand the impact of our algorithm on system performance.

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; only the channel-condition measurement mechanism need to be modified. Our attack can thus be performed using modified user equipment legitimately registered to a network.

The rest of our paper is organized as follows. In Section 2, we discuss the concept of our attack and perform case studies of various channel-aware networking protocols, evaluating through simulation the effect of a false channel condition reporting attacker. 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.

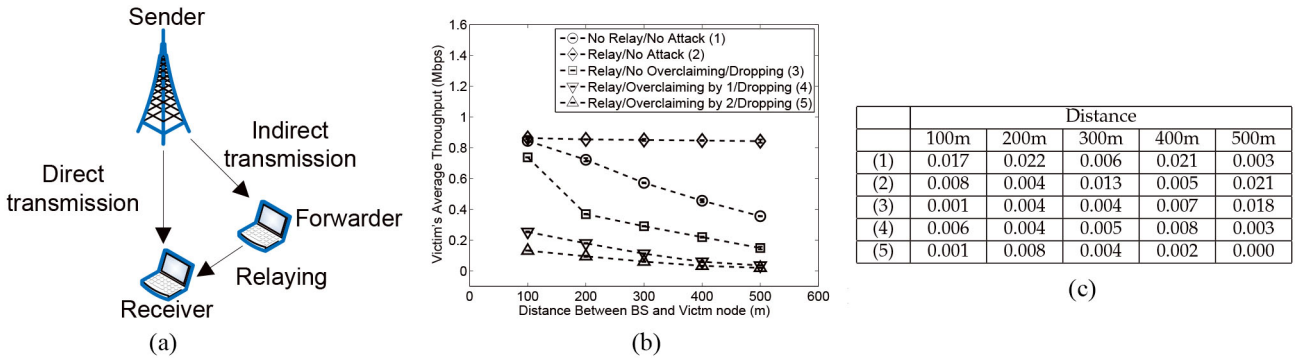
## 2 Attack

In this section, we introduce our attack concept and perform case studies to quantize the attack effects on specific channel-aware network protocols.

- *D. Kim is with HP Labs, Palo Alto, CA 94304. E-mail: dongho.kim@hp.com.*
- *Y.-C. Hu is with the Department of Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA. E-mail: yihchun@illinois.edu.*

*Manuscript received 8 Apr. 2011; revised 23 Feb. 2013; accepted 3 Aug. 2013. Date of publication 14 Aug. 2013; date of current version 15 May 2014. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier 10.1109/TMC.2013.104*

1536-1233 © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.




| | Distance | | | | |
|---|---|---|---|---|---|
| | 100m | 200m | 300m | 400m | 500m |
| (1) | 0.017 | 0.022 | 0.006 | 0.021 | 0.003 |
| (2) | 0.008 | 0.004 | 0.013 | 0.005 | 0.021 |
| (3) | 0.001 | 0.004 | 0.004 | 0.007 | 0.018 |
| (4) | 0.006 | 0.004 | 0.005 | 0.008 | 0.003 |
| (5) | 0.001 | 0.008 | 0.004 | 0.002 | 0.000 |

(c)

Fig. 1. Case Study of Cooperative Relaying Network. (a) Example network. (b) Attack effect. (c) Confidence intervals for each point of Fig. 1(b).

### 2.1 Attack Concept and Feasibility

Our study assumes that a network resource scheduler exploits channel condition information of each user to enhance network performance and each user reports its own channel condition to the scheduler. In this setting, a user can falsely report its channel condition. There are two different types of false reports: *underclaiming* (reporting its channel condition as worse than actual measurement) and *overclaiming* (reporting its channel condition as better than actual measurement). It is difficult to detect an underclaiming attack since for example, an attacker can selectively attenuate its capability, thus verifying its lower capability when tested. Hence, our paper considers only overclaiming attacks when we discuss our defense. However, our following case studies include the effectiveness of underclaiming attacks to provide complete understanding.

Depending on deployed PHY-layer technologies (e.g. OFDM), a system can utilize conditions for subchannels to perform more efficient frequency-selective scheduling [10]. Our work can apply for this case by handling each subchannel condition information separately. However, for clarity of presentation, we consider a single channel between network participants in this paper.

We can easily implement false channel condition reporting attack by modifying only a subcomponent to report channel condition. This subcomponent of a user equipment can be implemented in hardware or software. One recent trend of user equipment implementation is to increasingly move hardware part to software part for adaptable configuration of a general hardware [11]–[13]. The increasing software control of user equipment makes false channel condition reporting attack an increasingly practical attack.

### 2.2 Case Studies

We evaluate the effect of falsely reported channel condition under three types of channel-aware protocols: cooperative relaying protocols in hybrid networks, efficient routing metrics in wireless ad hoc networks, and opportunistic schedulers in high-speed wireless networks. For each protocol, we suggest possible attack mechanisms and quantify the effectiveness of the attack. We show that we can defend against some attacks using existing algorithms, and that other attacks require new security mechanisms. Each following case study has the same presentation format. First, we briefly explain the protocols. Then, we discuss effective attack scenarios for each protocol. Finally, we use simulations to evaluate the effect of each attack scenario.

#### 2.2.1 *Cooperative Relaying*

**Cooperative Relaying.** In a mobile wireless network, mobile nodes can experience different channel conditions depending on their different locations. When a node experiences a channel condition that is 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., [1], [2], [14], [15]) 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 [2], 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.

**Attack.** For simplicity, we discuss a single-hop relaying case even though our attack concept easily extends to multihop routes. Cooperative relaying architectures can take on various forms, but they all need to know the genuine channel condition of each candidate relaying node to find the most appropriate relaying node. The main purpose of cooperative relaying is to maximize system efficiency, so the route with best channel condition is likely to be chosen. When a node underclaims its channel condition, the node reduces its probability of being chosen for forwarding; the underclaiming attack is thus no worse than powering the node off. As a result, underclaiming is not an effective attack against cooperative relaying. We examine through simulation the effect of overclaiming attack. If an attacker overclaims its channel condition, the attacker is more likely to be chosen as the best candidate for relaying. Designating the attacker as a relaying node provides the attacker an opportunity to steal the packets or to adversely impact network performance.

**Evaluation.** We performed a simulation study to evaluate the overclaiming attack's effect on the normal users' performance in a cooperative relaying environment. We quantify

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 |

normal users' performance with and without the attack. We use the ns-2 simulator patched with EURANE [16], a UMTS system simulator. Our simulated network consists of one base station serving four users. The base station sends 11Mbps of Constant Bit Rate (CBR) traffic to each user. There is one attacker who may falsely report its channel condition to the base station. We represent channel condition using the Channel Quality Indicator (CQI) defined in the 3GPP standard [17].

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

Table 1 shows the transmission block size for each corresponding CQI. The same equation and block size are used for a case study of opportunistic scheduler presented in Section 2.2.3. We assume that one victim is close to the attacker so that when the victim experiences poor channel condition, the victim can use the attacker as a relaying node. The other two normal users get packets directly from the base station and do not participate in the cooperative relaying protocol. The victim and the two normal users honestly report their channel condition. The channel for the attacker and two normal users uses a shadowing plus Rayleigh model of a moving node 100m away from the base station with velocity of 3km/h. We vary the victim's distance to the base station from 100m to 500m to see the effect of the victim's channel condition on the performance degradation.

The attacker's goal in this simulation is to reduce the victim's throughput. The attacker can adopt two approaches. In the conservative approach, the attacker does not forward packets for the victim without falsely reporting its channel condition. In the aggressive approach, the attacker overclaims its channel condition so that the attacker can increase its probability of relaying packets for the victim.

Our simulations do not consider the overhead that an actual relaying protocol might incur in finding a new relaying node due to channel condition variation since such overhead is not related to the effect of attack. We assume that each transmission uses an orthogonal carrier so that transmissions do not interfere with each other. Our simulations do not implement a relay discovery protocol; rather, we compare the attacker and victim CQI, and use the link with better CQI value to transmit to the victim. This relaying scheme is an example; a system operator may choose a different scheme. However, the scheme that we chose is good for exploiting increased diversity to optimize throughput [2]. We ran each of our simulations for 100 simulated seconds.

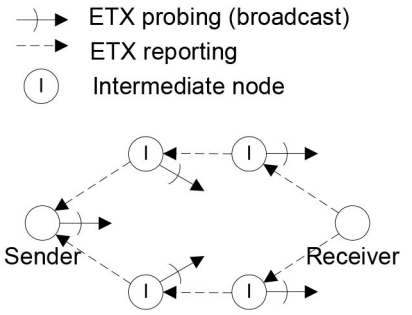
We measure the received throughput of the victim as shown in Fig. 1(b). 'No Relay/No Attack' represents the original UMTS system configuration with single hop transmission. 'Relay/No Attack' represents the attacker node relaying packets for the victim node whenever the attacker has a better channel condition than the victim node. We can see that when the victim uses the route with better condition, the system experiences improved throughput. 'Relay/No Overclaiming/Dropping' represents the attacker's conservative approach. As the victim node gets farther from the base station, the amount of decrement of the victim's throughput is larger due to the increased probability that the attacker's channel condition is better than the victim's channel condition. 'Relay/Overclaiming/Dropping' represents the case in which the attacker overclaims its channel condition, either by 1 or 2 CQI levels. Compared to the 'No Overclaiming' case, we can understand how much an attacker steal the chance to be chosen as a relayer by overclaiming. Our results show that even with a small amount of overclaiming, the attacker can steal many packets. Though we have only shown results for a single sample protocol, and an actual system may use a different protocol, we believe that the attack of falsely reporting the channel condition can have a harmful effect on the network even with a small amount of overclaiming.

#### 2.2.2 Efficient Routing Metrics in Wireless Ad Hoc Networks

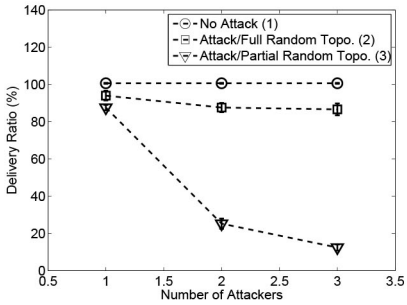
**Routing Protocols in Ad Hoc Networks.** A wireless ad hoc network supports communication between nodes without need for centralized infrastructure such as base stations or access points. To deliver packets to destinations out of a source node's transmission range, the source employs the help of intermediate nodes to forward each packet to its destination. Routing protocols in wireless ad hoc network discover routes between nodes (e.g. [18]–[20]). When there are multiple valid routes from a source to a destination, a routing protocol needs to choose among valid routes. A *routing metric* is a value associated to a route and represents the desirability of a route. A typical metric in the seminal routing protocols is minimum hop count. The rationale behind the metric of minimum hop count is that a route with fewer hops allows a packet to be delivered with the smaller number of transmissions. Hence, the metric of minimum hop count can reduce the total energy consumption across all network nodes as long as nodes do not perform power or rate adaptation.

However, the metric of minimum hop count may have an adverse effect on the number of transmissions due to the characteristics of wireless channel. Each hop in a route with minimum hop count is likely to cover a long distance. In a wireless channel, transmissions over longer distance are less likely to be successful. Hence, a route with minimum hop count can actually increase the number of transmissions. To address these shortcomings, researchers have proposed different routing metrics. One example is the *expected transmission count* (ETX) metric proposed by Couto *et al.* [3]. Their paper defines ETX as follows:

> The ETX of a link is the predicted number of data transmissions required to send a packet over that




(a)




(b)

| | Number of attackers | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| (1) | 0.463 | 0.463 | 0.463 |
| (2) | 5.208 | 4.936 | 6.378 |
| (3) | 2.942 | 5.516 | 3.318 |

(c)

Fig. 2. Case Study of Efficient Routing Metric. (a) Example network. (b) Attack effect varying Num. of attackers (Total six nodes). (c) Confidence intervals for each point of Fig. 2(b).

link, including retransmissions. The ETX of a route is the sum of the ETX for each link in the route.

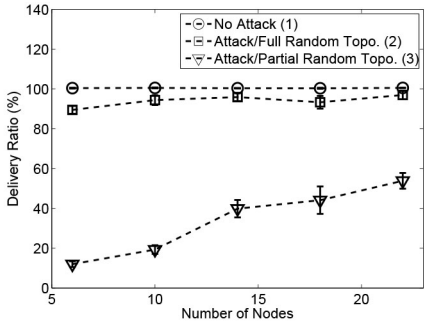
In a system using ETX, a node sends some probe packets to its neighbor nodes. The neighbor nodes report to the original node how many probe packets they receive. By comparing the number of probe packets received by each neighbor node to the total number of probe packets, a node estimates channel condition to each neighbor node. Specific implementation of ETX depends on routing protocols. We use ETX in DSR [19] for the following simulations though our attack concept is applicable to any routing metric and any routing protocol that is based on the channel condition of each link in the route.

**Attack.** To calculate the ETX of a route in forward direction, a source node needs to know the ETX of each forward link on the route. Each upstream node of each forward link reports its measured ETX to a source node. An intermediate node can overclaim by reporting a smaller ETX or underclaim by reporting a larger ETX. When an intermediate node underclaims, a source node is less likely to choose a route through the intermediate node. Hence, similarly to the case of cooperative relaying, an underclaiming node can be considered a power off node which may not attack a network. In contrast, an intermediate node that overclaims increases its probability of being chosen. Hence, an overclaiming node can maliciously intercept packets. The effectiveness of a false ETX report depends on the number of attackers in the route; a larger number of attackers on the same route can reduce the ETX further, thus having a greater impact on a source node's route selection.

**Evaluation.** We performed a simulation study to quantify the effect of falsely reported ETX in a wireless ad hoc network. We implemented measurement and reporting of ETX on DSR [19] in the ns-2 simulator; we chose DSR as a representative routing protocol, but our attack generalizes to any metric-based routing protocol. A detailed explanation of DSR is beyond the scope of our paper. We simulated a network where a source node sends CBR traffic at a rate of 1.31 kbps to a destination node and the remaining nodes forward packets. We use relatively low rate traffic to avoid performance degradation due to interference and to eliminate congestion as a factor in our results. We vary the number of attackers and measure the fraction of data packets successfully delivered. Each attacker claims that its ETX is 1, which is the strongest attack. If we use larger ETX, the attack strength will be reduced. Hence, our results show maximal attack effect. Each attacker drops any packets it receives for forwarding, so we can use the delivery ratio as a measure of the frequency with which the attacker is selected for forwarding. We consider two topologies; in the first, nodes are randomly placed in a plane of 600m by 600m, and in the second, we fix the location of the source and destination nodes 600m away from each other and randomly place the other nodes between them. We use shadowing model with ns-2 default parameters as our channel model. With these two topologies, we can see the effect of the topology on attack. We perform 100 runs for each type of topologies. For each run, we generate a different random topology. Fig. 2(b) shows the results with six nodes in the network. With the fully random topology, the delivery ratio remains about 90 percent, regardless of the number of attackers. With the partially random topology, the delivery ratio sharply decreases with an increasing number of attackers. The two topologies have differing results because an effective attack depends on two conditions. First, an attacker must be on a valid route from a source to a destination. Second, the fraction of intermediate hops that are attackers must be large enough for the attackers to substantially impact the ETX of the route. With the fully random topology, these two conditions are not likely to be satisfied because routes are relatively short (possibly one-hop at times). However, with the partially random topology results, well-located attackers can inflict significant harm to the network. As the number of attackers increases, the attackers are more easily able to control the ETX of a route. Fig. 3(a) shows the results with three attackers when we vary the number of nodes in the network. As the number of nodes increases, the attack effect is reduced, because dense legitimate nodes reduce the distance between nodes, thereby getting good quality channel conditions.

### 2.2.3 Opportunistic Schedulers

**Opportunistic Schedulers.** An *opportunistic scheduler* [5], [6] is a centralized resource scheduler that exploits the channel condition information of each user for efficient resource management. Channel condition variation due to fading in wireless networks induces different channel conditions for each user at each moment in time. This is called multiuser diversity. One simple example of an opportunistic scheduler is an efficiency-oriented scheduler that allocates




(a)

| | Number of nodes | | | | |
|---|---|---|---|---|---|
| | 6 | 10 | 14 | 18 | 22 |
| (1) | 0.4372 | 0.4529 | 0.2501 | 0.5185 | 0.2852 |
| (2) | 3.9306 | 4.9377 | 4.0305 | 6.4162 | 4.2822 |
| (3) | 2.8907 | 4.3355 | 8.7718 | 13.7759 | 7.9183 |

(b)

Fig. 3. Case Study of Efficient Routing Metric. (a) Attack effect varying Num. of nodes (Three attackers). (b) Confidence intervals for each point of Fig. 3(a).

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. Proportional Fair (PF) scheduler [5] is a widely known scheduler that addresses the fairness problem. When the PF scheduler collects channel condition from each user at time slot $t$, the PF scheduler determines the transmission rate $R_i(t)$ for each user $i$ accordingly. In our system, $R_i(t)$ is equal to block size determined by channel condition (shown in Table 1) divided by slot duration which is 2ms in our system. The PF scheduler determines whom to serve by comparing a metric $R_i(t)/T_i(t)$ for each user, where $T_i(t)$ is user $i$'s average throughput calculated as

$$\begin{cases} (1-1/t_c)T_i(t-1)+1/t_cR_i(t) & \text{if user } i \text{ is chosen} \\ (1-1/t_c)T_i(t-1) & \text{otherwise} \end{cases}$$

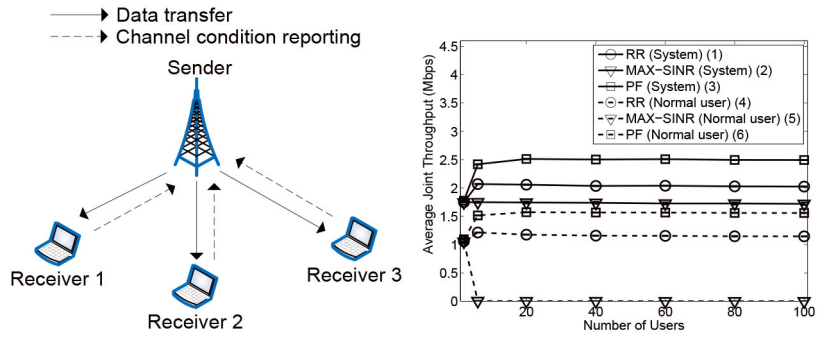
and $t_c$ represents time constant of a low pass filter. In each time slot, the PF scheduler serves the user with the largest metric. We consider the effects of the false channel condition reporting attack on these two schedulers, MAX-SINR and PF.

**Attack.** An attacker's objective is to steal as many time slots as possible. Against a MAX-SINR scheduler, the attacker can steal nearly all time slots simply by reporting the best possible channel condition. However, against a PF scheduler, the attacker is much more limited due to the fairness guarantee of the PF scheduler. With the PF scheduler, an attacker can steal specific time slots by overclaiming its channel condition to be the best possible condition. However, in future time slots later, the attacker is less likely to be chosen since the attacker's increased average throughput ($T_i(t)$) causes the attacker's metric ($R_i(t)/T_i(t)$) to decrease. Hence, even when overclaiming, the attacker under PF scheduler cannot occupy many consecutive slots. To obtain several consecutive time slots, an attacker can gradually increase the amount of overclaiming instead of overclaiming the best possible condition. However, as the attacker increases his claimed channel capacity, has $T_i(t)$ increases equally rapidly. Hence, it is difficult for an attacker under PF scheduler to impose a significant impact on network. This issue is also discussed by Racic *et al.* [7]. They maximize the effect of their false reporting attack by handover to intentionally reduce $T_i(t)$. This attack with handover is orthogonal to our work.

**Evaluation.** We performed a simulation study using opportunistic schedulers. We use ns-2 simulator patched with EURANE [21], a UMTS system simulator. 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 shadowing plus Rayleigh model modeling moving node with velocity of 3km/h. We sourced 11 Mbps of CBR traffic to each user. We simulate three scheduling policies: MAX-SINR, PF, and Round Robin (RR). RR gives each user the exact same amount of service opportunity by allocating the time slots to users in order. We vary the number of users and measure both the total system throughput and normal users' throughput for each of the three scheduling policies. In Fig. 4(b), the total system throughput is represented by the solid lines and the normal users' throughput is represented by the dotted lines. For MAX-SINR, the attack's effect is severe as expected. For an attacker to perform an effective attack, the attacker's packets should remain in the queue so that a scheduler serves an attacker instead of a normal user. When the number of users is small (e.g. two), the queue does not always hold attackers' packets, so the normal users get some throughput. When the number of users is larger than four, the attackers can occupy all throughput. For RR and PF, the normal users' throughput is about half of the system throughput. This result shows that falsely reporting CQI is not so harmful under RR and PF. This result coincides with the result of the work of Racic *et al.* [7].

### 2.3 Summary

We performed three case studies to study the attack effectiveness of falsely reporting channel condition. The simulation results show that in cooperative relaying networks and wireless ad hoc networks using ETX as a routing metric, the overclaiming attacks can effectively harm the normal users' service. For opportunistic schedulers, PF scheduler can effectively defend against overclaiming attackers. Considering the context of each protocol, we discussed the benefit of underclaiming action. We concluded that in the context of our case studies, underclaiming action does not give benefit to an attacker.




| | Number of users | | | | |
|---|---|---|---|---|---|
| | 20 | 40 | 60 | 80 | 100 |
| (1) | 0.008 | 0.008 | 0.008 | 0.011 | 0.007 |
| (2) | 0.037 | 0.035 | 0.037 | 0.041 | 0.041 |
| (3) | 0.007 | 0.003 | 0.008 | 0.004 | 0.004 |
| (4) | 0.005 | 0.004 | 0.007 | 0.009 | 0.005 |
| (5) | 0.001 | 0.000 | 0.000 | 0.000 | 0.000 |
| (6) | 0.004 | 0.002 | 0.003 | 0.003 | 0.003 |

(c)

Fig. 4. Case Study of Opportunistic Scheduler. (a) Example network. (b) Attack Effect. (c) Confidence Intervals for Selected Points of Fig. 4(b).

## 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 show our secure channel condition estimation algorithm.

### 3.1 Solution Spectrum

To defend against an attack that misreports the channel condition, we can use pre-existing mechanisms. One possible defense mechanism is anomaly detection [22]. Anomaly detection is a tool that monitors each user's performance to identify attackers. A response mechanism then disconnects the attacker from the network. Another possible defense mechanism is to mitigate the attack through fair allocation of network resources. Analogous to the PF scheduler for opportunistic schedulers, we can devise a mechanism that provides fairness guarantees regardless of reported channel condition for cooperative relaying networks or wireless ad hoc network routing.

Even though these approaches can mitigate the effectiveness of the attack, they have fundamental drawbacks. Anomaly detection mechanisms incur detection errors, which could result in incorrect termination of a normal user's service or failure to detect an attacker. When fairness is used to reduce the effect of the attack, we frustrate the original goal of cooperative relaying networks and ETX, which is to use resources most efficiently. An allocator considering fairness will substantially reduce the efficiency when compared to the original protocol, since fairness requires allocation of resources to less-capable channels.

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. In cooperative relaying networks and opportunistic schedulers, the base station can be a trustable entity. In wireless ad-hoc networks, the source node trying to establish a route to a destination is a trustable entity (because it trusts itself). 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 Secure Channel Condition Estimation

Our case studies and analysis of possible solutions in previous sections motivate a scheme that prevents an attacker from overclaiming its channel condition. In this section, we present our secure channel estimation scheme to prevent an overclaiming attacker. We do not consider underclaiming because in our case studies, an attacker gains no benefit from underclaiming, and because an attacker can always reduce its actual channel condition, for example by modifying his antenna. We start our presentation by giving intuition.

**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. In the following sections, we present our secure channel condition estimation algorithm in detail.

**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

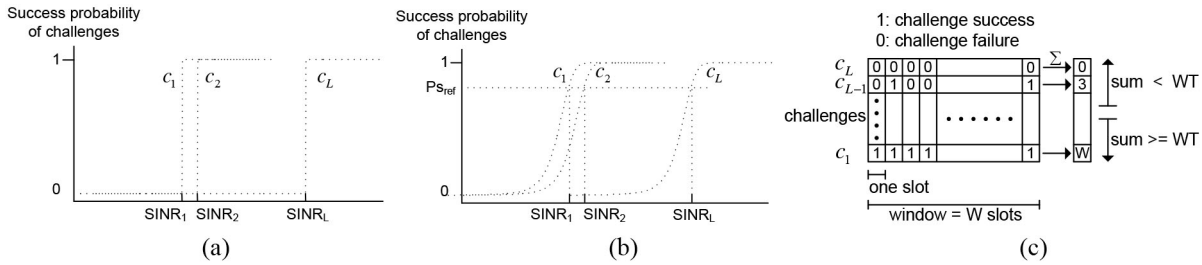



Fig. 5. Secure Channel Estimation. (a) Ideal challenge configuration. (b) General challenge configuration. (c) Channel estimation.

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 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. 5(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 $\text{SINR}_i \leq \text{SINR} < \text{SINR}_{i+1}$. In this case, we estimate the channel condition as $E_{i+1}$. These ideal challenges enable us to easily and accurately estimate the channel condition with only one time transmission of challenges. One way to construct challenges similar to ideal challenges is to use error correction codes (e.g., FEC). However, ideal challenges could require infinitely large challenges. Hence, our scheme considers non-ideal challenges, as shown in Fig. 5(b). For each challenge $c_i$, a node with channel condition as the threshold $\text{SINR}_i$ 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. 5(b), our scheme does not require the shape of each success probability to be the same. We discuss the design 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, in a practical point of view, a particular system may not provide various modulation and coding options. In such cases, we can vary transmission power accordingly.

**Transmission of Challenges.** The base station periodically sends a set of challenges to users. The period is one parameter to 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 size of the window 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. 5(c) shows the comparison procedure. We choose a threshold $T \in [0, 1]$. First, we see how any of the lowest rate challenges ($c_1$s) 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 $c_1$s 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 performance of our algorithm in terms of $T$ in Section 4.1.

## 4 Evaluation

In this section, we evaluate the performance and the security of our algorithm. Firstly, we analyze the performance of our algorithm according to algorithm parameters. This analysis can be used for parameter design guidelines. This analysis result is compared to simulation results. Secondly, we analyze the security of our algorithm. In this analysis, we show how much a brute-force attacker can be successful in guessing the value included in a challenge according to algorithm parameters. This analysis can be used for understanding tradeoff between security and system overhead. Thirdly, we integrate our algorithm into a network simulator and evaluate the effect of our algorithm on the system performance. We show that our algorithm securely and effectively estimates channel condition through most of its parameter space.

### 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 $E[|\text{CQI} - \widehat{\text{CQI}}|]$ according to algorithm parameters such as window size ($W$), threshold ($T$), the size of a challenge and $Ps_{ref(i)}$ of a challenge. CQI 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. Though this assumption does not hold in a mobile environment, the purpose of our analysis is not to capture every detail of real world but to verify our simulator. For the evaluation in a realistic environment, we perform simulations with channel models considering variable channel conditions, as described in Section 4.3. We assume that the challenge size and $Ps_{ref(i)}$ are the same for different challenges for easy comparison of performance. The equations in our analysis do not assume the same values of challenge size and $Ps_{ref(i)}$. However, with different values of challenge size and $Ps_{ref(i)}$, it is not easy to understand the parameters' effect on the performance. In this analysis, we assume that the challenges are authenticated. We consider attacks which try to forge a challenge in Section 4.5.

**Analysis.** Outline of our analysis is that given CQI, we calculate the probabilities that an estimated CQI ($\widehat{\text{CQI}}$) is determined to be each CQI-level 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 SINR. This function depends on the modulation and coding method used for constructing challenges, and is well-understood in communication theory [23]; 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

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

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 $P_{cs_i}$. 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. (The lowest CQI is 0 as in our previous case study.) 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

$$P_{ec}(i, \text{SINR}) = \left(\prod_{j=1}^{i} P_t(j)\right) \times \left(1 - P_t(i+1)\right),$$

where $P_t(i) = P_{c_i}(\lceil WT \rceil) + P_{c_i}(\lceil WT \rceil + 1) + \cdots + P_{c_i}(W)$. For CQI-level $L$, we have a little bit different form.

$$P_{ec}(L, \text{SINR}) = \prod_{j=1}^{L} P_t(j).$$

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

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

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 easy performance comparison. To obtain specific numerical results, we use the same CQI configuration as the UMTS system, which is explained in our study about cooperative relaying protocol in Section 2.2.1. Since $P_{ec}$ has a non-continuous function (ceil function), it is difficult to apply optimization theory. Hence, we evaluate the average estimation error exhaustively through the space of possible parameters. For that calculation, we use the successful reception probability of QPSK as $R_i(SINR, Ps_{ref(i)})$. We choose target SINR so that the space of overestimation and underestimation can be the same. Hence, we fix the target SINR as -1.19dB. The CQI corresponding to the target SINR is 15 which is a central point in a system having 31 different CQI values (0∼30).

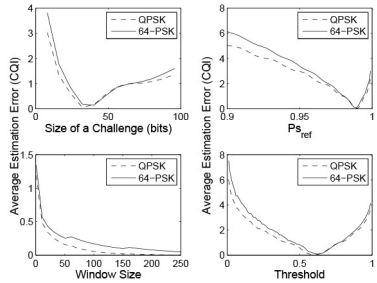



Fig. 6. Analysis on Average Estimation Error.

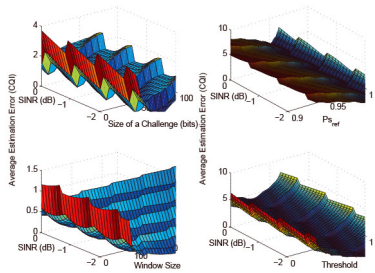



Fig. 8. Effect of SINR on Average Estimation Error.

Fig. 6 shows calculated average estimation error. Per each subplot, we vary one parameter keeping other parameters fixed. We use optimal values for other parameters. We used 32 bits as size of a challenge, 0.99 as $Ps_{ref}$, and 0.59 as threshold. We can see that the size of a challenge, reference probability of a challenge and threshold each has an optimal point. We illustrate why the optimal points exist by showing the probability that the number of successful challenge transmission is greater than the product of window size and threshold for each challenge according to various parameters in Fig. 7. As the size of a challenge increases, the probability curve slides to the underestimated point direction. The increasing threshold moves the probability curve in the same direction as the size of a challenge. The size of a challenge is also related to the security against a brute-force guessing attack, as analyzed in Section 4.5. With increasing reference probability of a challenge, the probability curve moves to the overestimated point direction. 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.

To understand how sensitive our analysis is to SINR, we varied SINR and examined the same set of numerical results as in Fig. 6. Fig. 8 shows that for various SINR-levels, the difference of our results is less than 1 in CQI.

### 4.2 Protocol Overhead

The overhead of our secure channel condition estimation scheme depends on assumptions. If we assume no collusion, a base station can broadcast challenges to users in a cell. This assumption is reasonable if we bound the time by which a user returns decoded challenges to base station to prevent another user to send its decoded challenge to the user.

When we implement our scheme, an important design issue is tradeoff between performance and overhead. In addition to the size of challenge shown in Fig. 6, the number of distinct channel conditions $(L + 1)$ affects the overhead of our scheme. While more challenges enable more precise estimation, they induce more overhead. Using our analysis in Section 4.1, we analyze the relationship between the number of challenges and performance. Fig. 9 shows average estimation error and overhead in terms of the number of challenges. Overhead represents how many bits a set of challenges is. We assume that the size of each challenge is 32 bits which represent optimal value shown in Fig. 6. We can see that 20 challenges results in good performance with comparatively small overhead.

### 4.3 Simulation

We performed a simulation study to verify our analysis and consider variable channel condition's effect on the performance. For system parameters (e.g. time slot duration), we use UMTS parameters since our simulator for evaluation of system performance in Section 4.6 is based on UMTS system. We start with the case of a static channel condition.

**Static Channel Condition.** We evaluate how accurately our algorithm estimates channel condition through simulation. We use the same mapping from channel condition to CQI as we use for previous case studies in Section 2.2. We choose the channel condition as -1.19dB which is a CQI of 15. For the construction of challenges, we assign a challenge for each of 30 CQIs. The challenge is modulated with QPSK. We simulate each challenge by shifting standard QPSK signal properly as described in Section 3. Modulation techniques affect the performance of our algorithm, which depends on how steep the success probability of challenges in terms of SINR is. For example, since the success probability curve of QPSK is steeper than that of 64-PSK, the performance with QPSK is better than 64-PSK as shown in Fig. 6. We use optimal values for $Ps_{ref}$ and the size of challenges that we obtain using analysis presented 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

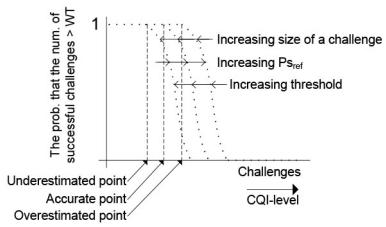



Fig. 7. Analyzing Parameter Design.

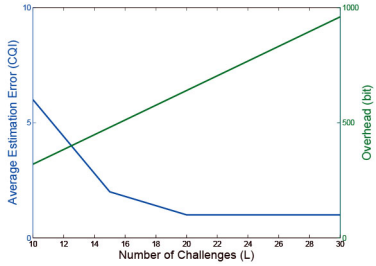



Fig. 9. Effect of Number of Challenges on Average Estimation Error.

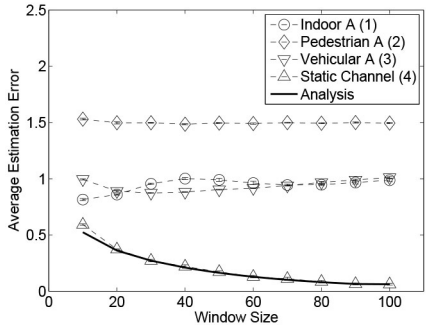



Fig. 10. Varying Window Size.

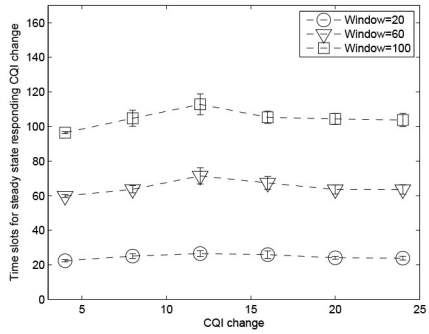



Fig. 12. Reaction time responding to CQI increment.

window size and threshold. 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. 10 and Fig. 11 show the average value of the differences. Simulation results match our analysis well. As window size increases, the average estimation error decreases as expected. For all values of window size, the estimation error is below 1. With window size of 100, the error is around 0.05. Even larger windows would 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 for an accurate estimation for static channel condition, the same parameter setting does not guarantee the same accuracy in variable channel condition. By using a variable condition channel model, we evaluate the effectiveness of our protocol under variable channel condition. With the same parameters of our algorithm as in our static channel condition evaluation, we use UMTS channel models such as Indoor A with velocity 3km/h, Pedestrian A with velocity 15km/h and Vehicular A with velocity 120km/h. Fig. 10 shows the effect of window size on average estimation error. The average estimation error in variable channel conditions is greater than the error in static channel condition, and the window size is not as a significant factor in accuracy as under a static channel condition; this shows that the variability of the channel condition limits the accuracy of our algorithm. Even then, in most cases, the error is not greater than 1. Furthermore, both legitimate nodes and attacking nodes experience similar errors, further reducing the effectiveness of overclaiming. Fig. 11 shows the average estimation error for various values of threshold. Again, the estimation error in 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 below 1.5, and that the optimal parameter for static channel condition applies to the case for variable channel condition cases.

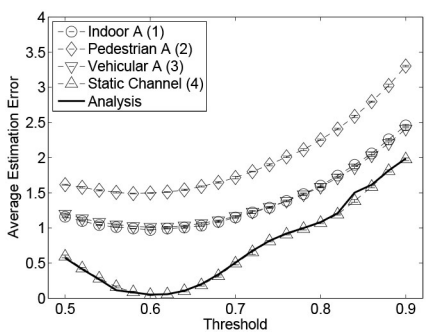



Fig. 11. Varying Threshold.

### 4.4 Reaction Time

To understand how quickly our secure channel estimation scheme responds to a change in channel condition, we performed a set of simulations having a step change of CQI level (e.g. CQI 0 to CQI 4). We measured the number of time slots before the estimated CQI reached steady state. Fig. 12 shows the required number of time slots vs. the amount of the change of CQI level for different window sizes of our estimation scheme. The main factor affecting reaction time is the window size: in almost all cases, the number of time slots is very close to the window size.

### 4.5 Security Analysis

In this section, we discuss possible attacks on our secure estimation of channel condition and corresponding defense mechanisms.

**Brute-force Attack.** The security of our secure estimation of 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 by brute-force search, or the attacker can attempt to decode the

TABLE 2
Confidence Intervals for Average Estimation Error

(a) Confidence Intervals for Each Point of Fig. 10

| | | (1) | (2) | (3) | (4) |
|---|---|---|---|---|---|
| Window size | 10 | 0.0186 | 0.0155 | 0.0108 | 0.0125 |
| | 20 | 0.0118 | 0.0212 | 0.0168 | 0.0092 |
| | 30 | 0.0109 | 0.0079 | 0.0064 | 0.0217 |
| | 40 | 0.0175 | 0.0083 | 0.0072 | 0.0310 |
| | 50 | 0.0276 | 0.0110 | 0.0074 | 0.0158 |
| | 60 | 0.0309 | 0.0124 | 0.0055 | 0.0215 |
| | 70 | 0.0142 | 0.0067 | 0.0082 | 0.0272 |
| | 80 | 0.0150 | 0.0077 | 0.0163 | 0.0212 |
| | 90 | 0.0253 | 0.0109 | 0.0134 | 0.0154 |
| | 100 | 0.0263 | 0.0109 | 0.0147 | 0.0153 |

(b) Confidence Intervals for Selected Points of Fig. 11

| | | (1) | (2) | (3) | (4) |
|---|---|---|---|---|---|
| Threshold | 0.5 | 0.0257 | 0.0105 | 0.0161 | 0.0415 |
| | 0.6 | 0.0171 | 0.0103 | 0.0145 | 0.0171 |
| | 0.7 | 0.0232 | 0.0375 | 0.0234 | 0.0426 |
| | 0.8 | 0.0416 | 0.0168 | 0.0200 | 0.0217 |
| | 0.9 | 0.0285 | 0.0212 | 0.0353 | 0.0371 |

received challenge value as a normal user would. Now, we will show that when the challenge values are chosen using a pseudo-random number generator, decoding is an attacker's dominating strategy. In other words, the attacker has no benefit to guess the challenge by brute-force.

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 $\sigma$. 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.

**Other Possible Attacks.** Until now, we have assumed that a challenge communicated between a base station and a receiver is valid. With some attacks like spoofing, main-in-the-middle, or replay attacks, an attacker can send invalid challenge to a user or send an invalid returned challenge to a base station. This invalid challenge can be used to make a base station believe that a receiver's channel condition is poorer than the actual channel condition. For example, suppose an attacker sends an invalid challenge a receiver, masquerading as the legitimate base station. When the receiver returns the invalid challenge to the base station, the base station will mistakenly believe that the receiver's channel condition is not good enough to decode the challenge. These attacks work when there is no authentication mechanism on the identity of a message originator or the time when a message is sent.

One way to handle these attacks is to use public key with CA (Certificate Authority). When a user connects to a base station, the user gets a public key using pre-installed public key of a CA. Pre-installed public key of a CA prevents an attacker from performing a man-in-the-middle attack. To defend against spoofing and replay attack, a sender can attach a digest of challenges and a timestamp. Design of these defense mechanisms are orthogonal to our work.

### 4.6 System Performance

So far, we have evaluated the performance and the security of our secure channel estimation algorithm itself. Now, we evaluate how much our secure channel condition estimation algorithm enhances the 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 use for the previous case studies of cooperative relaying system and opportunistic scheduler system in Section 2.2. We do not evaluate the system performance for efficient routing metric in wireless ad hoc networks since we need an additional protocol for multi-hop forwarder case. (We consider only a single hop relayer in cooperative relaying network case.) In multi-hop forwarder case, intermediate forwarder nodes should integrate our algorithm and report their estimated channel condition to a source node. However, since we cannot trust intermediate nodes' report, we need an additional protocol to prevent intermediate nodes from cheating on the estimated channel condition report. Such a protocol development is beyond the scope of this paper. We leave this protocol as a future work.

**Cooperative Relaying Network.** We implemented our algorithm in ns-2 simulator. Our basic simulation setting is the same as our case study for a cooperative relaying network. In the network, the base station plays a role as a source for traffic. A victim node can choose a relayer node with better channel condition. An attacking relayer node overclaims its channel condition to intercept packets to the victim node. As shown in Section 4.3, a channel model can affect the estimation error of our algorithm. Hence, we use three different channel models (Indoor A, Ped A, and Veh A) that we used before for the simulation of variable channel condition to understand how much the estimation error due to variability of channel condition affects the system performance. As we vary the distance between the base station and the victim node, we measure the throughput that the victim node can get under false channel condition reporting attack.

Fig. 13 shows the measured results in the case of a single relayer for the victim node. In this simulation, the attacker node is the only relayer for the victim node. We consider three different cases: overclaiming by 1, overclaiming by 10 and defense with our algorithm. For the cases of overclaiming by 1 and 10, 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. 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. We performed simulations for two-relayers case where there is a legitimate relayer; in the simulations, attack effect is reduced due to the legitimate relayer. Other than the improved performance, the same trends exist as in Fig. 13.

**Opportunistic Scheduler.** As in the case of cooperative relaying, we implemented our algorithm in the simulation environment that we used for the case study of

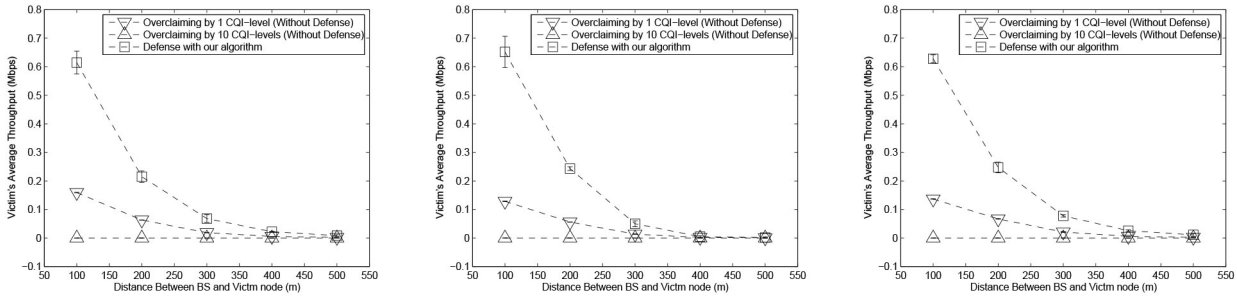



Fig. 13. Effect of Our Algorithm on A Relaying Network Example (One Relayer). (a) Indoor A. (b) Ped A. (c) Veh A.

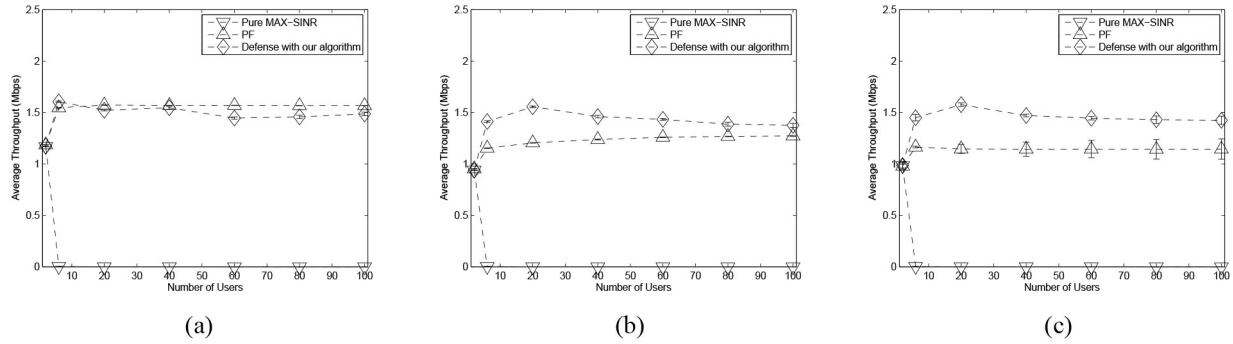



Fig. 14. Effect of Our Algorithm on Opportunistic Scheduler. (a) Indoor A. (b) Ped A. (c) Veh A.

opportunistic scheduler. We used identical parameters, except we replaced the static channel with various variable channel models. We measure the throughput of normal users under scheduling policies of MAX-SINR, PF and MAX-SINR with our algorithm. In MAX-SINR with our algorithm, a base station does not use reported CQI-level to determine a user with the best channel condition in a give time slot. Instead, the base station uses CQI-level estimated by our algorithm. In the case study of opportunistic scheduler, our observation was that PF scheduler prevented attackers from stealing throughput. Hence, we concluded that PF was a good candidate for defending against false channel condition reporting attack. However, our simulation results for the system performance show that MAX-SINR with our algorithm can achieve higher throughput than PF scheduler in most cases. The fact that the performance of our algorithm depends on channel characteristic affects the throughput of normal users in case of MAX-SINR with our algorithm.

## 5 Related Work

**Attacks on Hybrid Networks.** Carbunar *et al.* [14] propose JANUS, a hybrid network architecture implementing cooperative relaying. They state the possibility of a rate inflation attack in which a node reports larger bandwidth to base station than the node can provide. JANUS addresses rate inflation by having a base station send request packets, which nodes must acknowledge. JANUS exploits the fact that a rate-inflation attacker's link can induce excessive packet losses. Their approach does not consider the probabilistic effects of a wireless channel, and when an attacker can acknowledge some of the probe packets, the base station cannot detect the attacker. As a result, the authors admit that JANUS does not prevent a rate inflation attack. Haas *et al.* [15] propose SUCAN defending against Byzantine behaviors in hybrid networks. However, their attack behaviors do not include cheating on channel condition.

**Attacks on Routing Protocols in Ad-hoc Networks.** In wireless ad-hoc routing protocols, an attacker can inject forged routing packets into a network to make routing disruption. Many researchers studied defense mechanisms against such attacks [24]. However, there is no study on false channel condition reporting attack against efficient routing metrics.

**Attacks on Opportunistic Schedulers.** Bali *et al.* [9] 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.* [7] on PF scheduler is the closest work to ours in the sense that they consider the attack 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 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 setting.

## 6 Conclusion

In this paper, we have studied the threat imposed by falsely reporting users' channel condition. Through case studies for three different types of wireless network protocols, we

show that in a cooperative relaying network and a network using ETX, a false reporting attack can significantly reduce the performance of other users. 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.

## ACKNOWLEDGMENTS

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

## REFERENCES

[1] A. Sendonaris, E. Erkip, and B. Aazhang, "Increasing uplink capacity via user cooperation diversity," in *Proc. IEEE ISIT*, Cambridge, MA, USA, Aug. 1998, p. 156.
[2] H. Luo, R. Ramjee, P. Sinha, L. E. Li, and S. Lu, "Ucan: A unified cellular and ad-hoc network architecture," in *Proc. ACM MobiCom*, San Diego, CA, USA, 2003, pp. 353–367.
[3] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *Proc. ACM MobiCom*, San Diego, CA, USA, 2003, pp. 134–146.
[4] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," in *Proc. ACM SIGCOMM*, Portland, OR, USA, 2004, pp. 133–144.
[5] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *Proc. IEEE VTC*, Tokyo, Japan, 2000, pp. 1854–1858.
[6] P. Viswanath, D. N. C. Tse, and R. Laroia, "Opportunistic beamforming using dumb antennas," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1277–1294, Jun. 2002.
[7] R. Racic, D. Ma, H. Chen, and X. Liu, "Exploiting and defending opportunistic scheduling in cellular data networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 5, pp. 609–620, May 2010.
[8] U. Ben-Porat, A. Bremler-Barr, H. Levy, and B. Plattner, "On the vulnerability of the proportional fairness scheduler to retransmission attacks," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 2011, pp. 1431–1439.
[9] S. Bali, S. Machiraju, H. Zang, and V. Frost, "A measurement study of scheduler-based attacks in 3G wireless networks," in *Proc. PAM*, Berlin, Germany, 2007.
[10] F. Wang *et al.*, "IEEE 802.16e system performance: Analysis and simulations," in *Proc. IEEE PIMRC*, Berlin, Germany, Sept. 2005, pp. 900–904.
[11] *Odyssey 8500* [Online]. Available: http://www.wavesat.com/pdf/OD-8500-IC-PB.pdf
[12] *Airspan* [Online]. Available: http://www.airspan.com/products_wimax.aspx
[13] J. Mitola, "The software radio architecture," *IEEE Commun. Mag.*, vol. 33, no. 5, pp. 26–38, May 1995.
[14] B. Carbunar, I. Ioannis, and C. Nita-Rotaru, "Janus: A framework for scalable and secure routing in hybrid wireless networks," *IEEE Trans. Dependable Secur. Comput.*, vol. 6, no. 4, pp. 295–308, Oct./Dec. 2009.
[15] J. J. Haas and Y.-C. Hu, "Secure unified cellular ad hoc network routing," in *Proc. IEEE Globecom*, Honolulu, HI, USA, 2009.
[16] *EURANE : Enhanced UMTS Radio Access Network Extensions for ns-2* [Online]. Available: http://eurane.ti-wmc.nl/eurane/
[17] *Physical Layer Procedures (FDD), Release 5*, V5.5.0, 3GPP Organizational Partners Technical Specification Group Radio Access Network 25.214, Jun. 2003.
[18] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. ACM SIGCOMM*, London, U.K., 1994.
[19] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, Boston, MA, USA: Kluwer Academic, 1996, pp. 153–181.
[20] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. WMCSA*, New Orleans, LA, USA, Feb. 1999, pp. 90–100.
[21] J. Antoniou, V. Vassiliou, A. Pitsillides, G. Hadjipollas, and N. Jacovides, "A discrete event based simulation environment for enhanced UMTS 3rd generation networks," in *Proc. 2004 ACM SAC*, New York, NY, USA.
[22] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009.
[23] J. Proakis, *Digital Communications*, 4th ed. Boston, MA, USA: McGraw-Hill, 2000.
[24] Y.-C. Hu and A. Perrig, "A survey of secure wireless ad hoc routing," *IEEE Security Privacy*, vol. 2, no. 3, pp. 28–39, May/Jun. 2004.

**Dongho Kim** received the B.S. and M.S. degrees in electrical engineering in 1999 and 2005, respectively, from Korea Advanced Institute of Science and Technology, Daejeon, South Korea, and the Ph.D. degree in computer engineering in 2012 from the University of Illinois at Urbana-Champaign, Urbana, IL, USA. His Ph.D. dissertation focused on secure resource management in networks. He is a research engineer at HP Labs, Palo Alto, CA, USA. Before joining HP Labs, he was a software engineer at Cisco Systems. His current interest includes security in mobile networks and mobile applications.

**Yih-Chun Hu** received the B.S. degree in computer science and pure mathematics from the University of Washington, Seattle, WA, USA, in 1997, and the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburgh, PA, USA, in 2003. He is an Associate Professor with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA. In his thesis work at Carnegie Mellon University, he focused on security and performance in wireless ad hoc networks. After receiving the Ph.D. degree, he has been with the Post-Doctoral Researcher with the University of California, Berkeley, doing research in the area of network security. His research interests include systems and network security.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.