Hearing Is Believing: Detecting Wireless Microphone Emulation Attacks in White Space

Shaxun Chen, Kai Zeng, and Prasant Mohapatra, *Fellow*, *IEEE*

Abstract—In cognitive radio networks, an attacker transmits signals mimicking the characteristics of primary signals, in order to prevent secondary users from transmitting. Such an attack is called primary user emulation (PUE) attack. TV towers and wireless microphones are two main types of primary users in white space. Existing work on PUE attack detection only focused on the first category. For the latter category, primary users are mobile and their transmission power is low. These properties introduce great challenges on PUE detection and existing methods are not applicable. In this paper, we propose a novel method to detect the emulation attack of wireless microphones. We exploit the relationship between RF signals and acoustic information to verify the existence of wireless microphones. The effectiveness of our approach is validated through real-world implementation. Extensive experiments show that our method achieves both false positive rate and false negative rate lower than 0.1 even in a noisy environment.

Index Terms—Primary user emulation attack, cognitive radio, wireless microphone, white space

1 Introduction

The popularity of wireless communication and the ever-increasing wireless traffic have put significant pressure on spectrum utilization. Recognizing the significance of spectrum shortage, the Federal Communications Commission (FCC) released analogue TV bands, often referred to as white space, to unlicensed users on a noninterference basis.

To access white space, unlicensed users (secondary users) must, according to FCC's rules, sense the spectrum before transmitting and evacuate immediately when a licensed user (primary user) appears in the same band. The most important and commonly seen primary users in white space are TV towers and wireless microphones, which have the priority over any secondary users.

While such shared-style spectrum accessing increases the efficiency of spectrum utilization, it introduces a new type of attack: primary user emulation (PUE) attack [1]. In such an attack, an adversary transmits signals whose characteristics emulate those of primary users, thereby causing legitimate secondary users to erroneously identify the attacker as a primary user. The goal of such attackers is either selfishly maximizing its own spectrum usage or just maliciously preventing other secondary users from communicating (in the latter case, it is a type of DoS attack).

PUE attack is not very difficult to launch since cognitive radios are highly reconfigurable with their software-based air interface. Because of the priority basis of white space access, legitimate secondary users are very susceptible to this type of attack.

- *S. Chen and P. Mohapatra are with the Department of Computer Science, University of California, Davis, Davis, CA 95616. E-mail: sxch@ucdavis.edu, prasant@cs.ucdavis.edu.*
- *K. Zeng is with the Department of Computer and Information Science, University of Michigan-Dearborn, 4901 Evergreen Rd., 244 CIS, Dearborn, MI 48128. E-mail: kzeng@umich.edu.*

Manuscript received 22 Feb. 2011; revised 16 Oct. 2011; accepted 2 Dec. 2011; published online 22 Dec. 2011.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2011-02-0091. Digital Object Identifier no. 10.1109/TMC.2011.272.

On the other hand, the detection of PUE attacks is a nontrivial task, because FCC requires "no modification to the incumbent (primary) system should be required to accommodate opportunistic use of the spectrum by secondary users" [2]. This implies that primary signals just stay what they were; secondary users must rely on their own to sense primary signals and differentiate emulation attackers. In other words, conventional approaches, such as embedding signatures in primary signals or employing an interactive protocol between a primary user and secondary users, cannot be applied to defend PUE attacks.

Several advanced approaches have been proposed for mitigating PUE attacks [1], [3], [4], [5], [6]. However, all of them focus on the attackers that emulate stationary primary users (TV towers). They are based on the fact that the locations of TV towers are fixed and assume that these locations are preknown by secondary users. Some solutions also make use of the feature that attackers can only emulate signal characteristics of TV towers in a relatively small area or direction. That is because a PUE attacker cannot emit the signal as strong as a genuine TV tower, which is typically very high and have tremendous transmission power.

In white space, there is another important category of primary users: wireless microphones (actually much greater in number than TV towers), which are neither stationary nor highly powered. They are easier to be emulated by adversaries, and the existing solutions cannot be applied. Detecting emulation attacks of wireless microphones is a much harder problem because of their variable locations and low transmission power.

In this paper, we proposed a novel method to detect wireless microphone emulation attacks. In our approach, each secondary user is equipped with an acoustic sensor. Relationship between energy level of RF signals and acoustic information received by the sensor are exploited to verify the authenticity of wireless microphones.

To the best of our knowledge, this is the first work dealing with emulation attacks of wireless microphones in

white space. In addition, our method does not require complex hardware. We demonstrate the effectiveness of our approach through experiments, which show that our method achieves both false positive rate and false negative rate lower than 0.1 even in a noisy environment.

The rest of this paper is organized as follows: Section 2 discusses related work. Section 3 defines the problem, and Section 4 presents our method detecting wireless microphone emulation attacks. In Section 5, we extend our work to a noisy environment by collaborative sensing. Section 6 evaluates our work in real-world settings. Section 7 discuses several related issues and Section 8 concludes the paper.

2 Related Work

Many signal detection techniques have been proposed and they mainly fall into three categories: matched filtering, energy detection, and cyclostationary feature detection [9]. Among them, energy detection has relatively low hardware requirement. Most radio devices are able to measure the energy level of RF signals without any hardware modification. In cognitive radio networks, signal detection techniques are used to detect primary users, but conventionally, they are researched in nonadversary environments [10], [11], [12].

Chen and Park first proposed two location-based approaches to detect PUE attacks [1]. The first is *distance ratio test*. It is based on the two-ray ground reflection model, which says the signal strength received from the primary user should be inversely proportional to the distance to the fourth power. Under such an assumption, the location of primary signal source can be determined if two or more secondary users exchange their power strength measurements. Then, the calculated location is compared with the TV tower map to judge if it is a genuine primary user. This method is very vulnerable to signal strength fluctuation. The other method is called *distance difference test*. It utilizes synchronization pulses embedded in analog TV signals to calculate the location of a primary signal, which requires strict time synchronization among secondary users.

Chen et al. also proposed another signal strength-based method which is more tolerant to signal fluctuation [3], [4]. This method relies on an underlying wireless sensor network which covers a fairly large area. Each sensor measures the signal strength of the primary user. Then, local averaging smooth technique is applied and signal strength geo-peaks are assumed to be the location of primary transmitters.

All these above methods try to locate the signal source to detect fake primary users. The locations of genuine primary users must be known a priori. Therefore, they cannot be used when the primary user is mobile or their locations are unpredictable.

Jin et al. presented a theoretical analysis on the distribution of received power, in order to differentiate an attacker from a genuine primary user [5], [6]. A Wald's sequential probability ratio test is employed to ensure a decent false positive and false negative. However, they assume that the primary users must be far away from all secondary users, and genuine primary users and emulation attackers subject to different propagation models (genuine TV towers have much higher antennas and stronger transmission power than attackers). These assumptions may be true in the context of TV broadcasting, but they do not hold when the attacker is trying to mimic the signal of wireless microphones.

Recognizing the difficulty of detecting wireless microphone emulation attacks, the 802.22 Task Group 1 proposed the disabling beacon protocol [7], [8], which suggests transmitting a specially designed signal before starting wireless microphones. If additional information, such as signatures, is embedded into the beacon, this method can help secondary users to differentiate genuine wireless microphones from attackers. However, while newly built wireless microphones can choose to incorporate the beacon protocol, there are still a great number of legacy wireless microphone users. Considering the fact that most of these users have not even registered their wireless microphones, we cannot expect that they will be equipped with a separate beacon device in the near future.

A preliminary version of our work was reported in [18]. In this paper, we further extend our approach to noisy environments by employing collaborative sensing. We introduce noise mitigation techniques to make our method practical in real-world scenarios.

3 Problem Definition

3.1 Preliminaries

As mentioned in Section 1, wireless microphones are primary users in white space. They are widely used in live performances, university lectures, sporting events, etc. In some musical theatre productions, even dozens of radio microphones work together.

Wireless microphones typically operate in VHF or UHF bands. According to FCC's regulation, they should occupy the bandwidth no more than 200 kHz (the same as a frequency modulation (FM) radio or analogue TV channel), and the power output is limited to 250 mW or less on UHF or 50 mW on VHF. In practice, this value is typically 10 to 50 mW due to battery life considerations.

Most wireless microphones use FM [13]. In the following methods and algorithms, FM wireless microphones are assumed for representativeness. The transmission range of wireless microphones' RF signal is usually less than 100-150 meters.

The working process of a wireless microphone is shown in Fig. 1. The microphone transforms the sound wave $\mathrm{w}(t)$ into current signal $\mathrm{m}(t)$, which has the same characteristics of the original sound wave: the amplitude stands for the loudness and the frequency shows the pitch. After that, the current signal is modulated by FM, and the modulated signal $\mathrm{S}(t)$ is sent into the air.

In the receiver end, the demodulator listens to $\mathrm{S}(t)$, and demodulated it into current signal $\mathrm{m}'(t)$. Then, $\mathrm{m}'(t)$ is output to loudspeakers or power amplifiers. The regenerated sound wave is noted as $\mathrm{w}'(t)$.

3.2 Attack Model and Problem Description

The emulation attacker mimics the characteristics of a wireless microphone's signal, in order to make secondary users erroneously identify it as a primary user. Since wireless microphones have low RF power, and can be turned on at

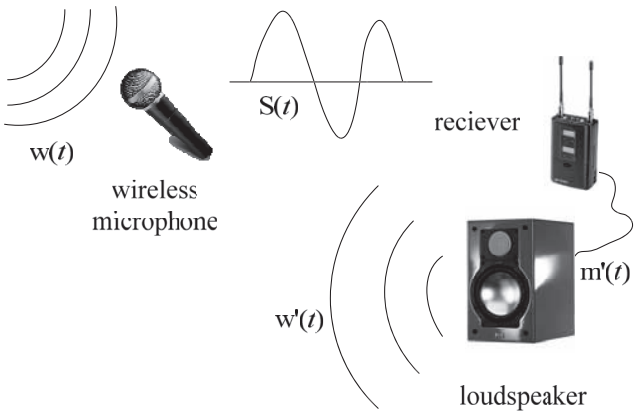


Fig. 1. Signal flow of a wireless microphone.

any time anywhere, they are relatively easier to be emulated. We assume that the attacker has full capability to emulate wireless microphones' transmission power, modulation type, bandwidth occupation, and any other characteristics of $S(t)$.

However, the attacker usually does not emit sound wave (to emulate $w(t)$ or $w'(t)$), because in that way it will be very easily detected out. More detailed discussion is included in Section 7.1.

The problem to be solved is differentiating emulation attackers from genuine primary users (wireless microphones). We have discussed the incapability or limitations of existing methods in Section 2. For our method, each secondary user is equipped with a sound sensor. We identify a genuine wireless microphone by exploiting the relationship between RF signals received by the secondary user and the environmental sound captured by its sound sensor (noted as $w''(t)$). If the signals do not pass our correlation test, an emulation attack is assumed.

Incorporating acoustic information opens a new window for small-scale, mobile primary user detection, but there are still substantial challenges:

1. *Correlating sound to the energy level of RF signal.* This is simple for amplitude modulation (AM), but wireless microphones use FM, where the relationship is relatively difficult to exploit. We choose to use energy detection because it is fast and simple. All cognitive radio devices are able to detect signal power without hardware modification.

   A straightforward alternative is to demodulate $S(t)$ into sound, and then compare it with $w''(t)$. However, this means, we have to add FM demodulators and rebuild the secondary users' internal circuit. More important, although they all use FM, different wireless microphones have different signal formats, such as mono, stereo, bandwidth, companding techniques, etc. It is very difficult for a single device to demodulate various wireless microphones from different manufacturers.
2. *Detecting attackers in a noisy environment.* $w''(t)$ is different from $w(t)$ especially when there are other acoustic sources nearby. How to reduce false positives when detecting emulation attacks is a challenging problem.
3. *Timing constraint.* The 802.22 standard draft specifies that sensors must be able to detect wireless microphone signals over a 200 kHz band within 2 seconds with both false-alarm and misdetection probabilities less than 0.1. Therefore, fast detection of emulation attacks is highly desirable.

We will focus on these challenges in Sections 4 and 5.

4 Detecting Emulation Attack of Wireless Microphone

In this section, we present our methodology for defense against wireless microphone emulation attacks. We first infer the relationship between $w(t)$ and $S(t)$, and then describe how to use this relationship to detect emulation attacks.

4.1 Relationship between Acoustic Signal and FM Power

Let $w(t)$ be the sound wave. The microphone transforms it into current signal $m(t)$. Ignoring the nonlinear distortion, we have

$$m(t) = \alpha w(t), \tag{1}$$

where α is a constant. After frequency modulation, the modulated signal is

$$S(t) = A_c \cos\left[2\pi f_c t + 2\pi k_f \int_0^t \alpha w(\tau) d\tau\right], \tag{2}$$

where A_c is the carrier amplitude, f_c is the carrier frequency, and k_f is the sensitivity of the modulator.

Without loss of generality, let $\alpha w(t) = A_m \cos(2\pi f_m t)$, and substitute it into (2):

$$S(t) = A_c \cos\left[2\pi f_c t + \frac{k_f}{f_m} A_m \sin 2\pi f_m t\right]. \tag{3}$$

Let $m_f = k_f * A_m / f_m, \omega_c = 2\pi f_c$, and $\omega_m = 2\pi f_m$, we have

$$\begin{aligned} S(t) &= A_c \cos(\omega_c t + m_f \sin \omega_m t) \\ &= A_c[\cos \omega_c t \cos(m_f \sin \omega_m t) - \sin \omega_c t \sin(m_f \sin \omega_m t)] \end{aligned} \tag{4}$$

and noting the trigonometric relationship that

$$\cos(m_f \sin \omega_m t) = J_0(m_f) + \sum_{n\ even}^{\infty} 2J_n(m_f) \cos(n\omega_m t)$$

$$\sin(m_f \sin \omega_m t) = \sum_{n\ odd}^{\infty} 2J_n(m_f) \sin(n\omega_m t),$$

where J_n are Bessel functions of the first kind, of order n. Substituting these two expressions into (4), we have

$$\begin{aligned} S(t) = {} & A_c J_0(m_f) \cos \omega_c t \\ & + \sum_{n\ even}^{\infty} A_c J_n(m_f)[\cos(\omega_c + n\omega_m)t + \cos(\omega_c - n\omega_m)t] \\ & + \sum_{n\ odd}^{\infty} A_c J_n(m_f)[\cos(\omega_c + n\omega_m)t - \cos(\omega_c - n\omega_m)t]. \end{aligned} \tag{5}$$

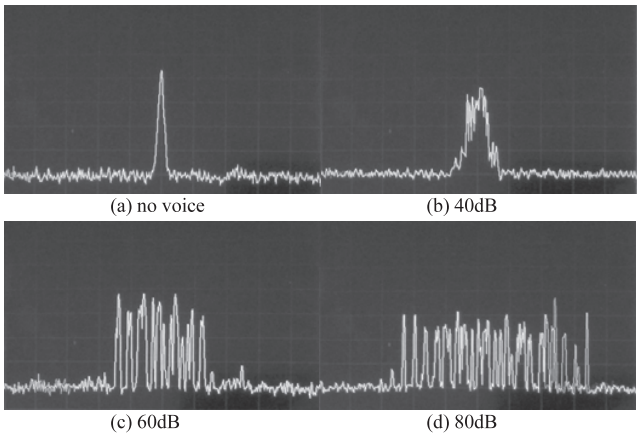


Fig. 2. Power distribution at different acoustic volume.

Equation (5) shows the frequency components of $\mathrm{S}(t)$. Besides f_c, it also has frequency components on $f_c \pm nf_{\mathrm{m}}(\mathrm{n} \in \mathrm{Z}^+)$, which are called side bands. According to the property of Bessel functions, when x goes larger, the values of $J_o(x), J_1(x), J_2(x)$, etc., become closer, which means more side bands significantly contribute to the power of $\mathrm{S}(t)$. Therefore, the power around center frequency (carrier frequency) reduces when m_f increases, because the total power contained in an FM wave is constant [14].

As we know, k_f is a constant and $m_f = {k_f}^* A_{\mathrm{m}}/f_{\mathrm{m}}$. We refer to the power of $\mathrm{S}(t)$ within $[f_c - \Delta f, f_c + \Delta f](\Delta f \ll$ bandwidth of $\mathrm{S}(t))$ as $P_{2\Delta f}$ thereinafter. Therefore, we come to our conclusion:

Lemma. *$P_{2\Delta f}$ reduces when $\frac{A_m}{f_m}$ goes up, and vice versa.*

This conclusion applies to any FM wireless microphone, no matter it is mono or stereo, with or without companding techniques. This is one of the reasons that we choose to detect signal power instead of converting $\mathrm{S}(t)$ back to sound. The only concern is that Δf should be always smaller than the bandwidth of the wireless microphone.

4.2 Emulation Attacker Detection

As derived in Section 4.1, $P_{2\Delta \mathrm{f}}$ is related to both the frequency and amplitude of sound wave. Between them, the amplitude of sound wave is the key factor. The reason is as follows.

First, the frequency range of human voice is from 300 to 3,000 Hz, where the upper bound 10 times the lower bound. However, loudness varies more. A normal speaker has the sound level about 40 to 80 dB, and lower than 20 dB when pauses. The power of 80 dB is 10^6 times higher than that of 20 dB, and 1,000 times larger in terms of amplitude.

Second, the frequency of a speaker usually does not change much, but the loudness can change suddenly because voice has pauses and emphases.

Therefore, loosely speaking, larger is the A_{m}, smaller is the $P_{2\Delta \mathrm{f}}$. Fig. 2 illustrates the power distribution of $\mathrm{S}(t)$ over the spectrum. We replay a piece of recorded sound at different volumes as the acoustic source of a wireless microphone, and measure the RF signal by a spectrum analyzer. As the figure shows, when the volume gets higher, more power spreads to the side bands in FM modulation. That is, less power is left in the central frequency (smaller $P_{2\Delta \mathrm{f}}$). We also record sound with different frequencies, but

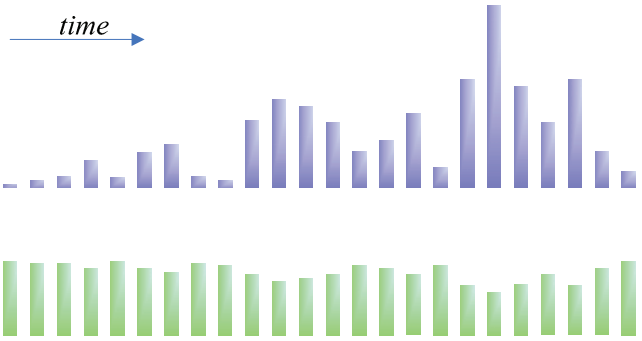


Fig. 3. Relationship between sound and RF signal.

power distribution of FM signals does not change much due to frequency variance.

Fig. 3 further illustrates the relationship over time between A_m and $P_{2\Delta \mathrm{f}}$ of a genuine wireless microphone. The upper bars are amplitudes of the sound wave while the lower bars indicate $P_{2\Delta \mathrm{f}}$. We can see when the sound is louder, the $P_{2\Delta \mathrm{f}}$ goes lower, which agrees with our theoretical analysis.

Now, we describe our method for emulation detection. If a wireless microphone signal is detected, a secondary user immediately performs the operations as follows to determine if it is an emulation attacker.

First, the secondary user shrinks its radio bandwidth to $2\Delta f$, with the center frequency unchanged (the same as the center frequency of the band where the wireless microphone is detected). $2\Delta f$ is set to 25 kHz by default. Because most wireless microphones have a RF bandwidth of 100 or 200 kHz, too large Δf makes it hard to capture the power variance around center frequency. On the other hand, smaller Δf can improve the performance of our method, but some cognitive radio devices may not have such narrow band-pass filters.

Then, the secondary user synchronizes the acoustic signal with the RF signal, and samples $\mathrm{w}''(t)$ with its sound sensor. Every Δt, it calculates the average amplitude of the samples. At the same time, it measures the average $P_{2\Delta \mathrm{f}}$ every Δt, and evaluates the relationship between $P_{2\Delta \mathrm{f}}$ and averaged amplitudes. The algorithm is given as follows:

Algorithm 1. Emulation attack detection

```
score = 0;
lastAvgAmp = lastAvgPwr = 0;
diff = signalSync();
wait(diff);
for (i = 0; i < n; i++)
    time = getCurrentTime();
    wait (Δt);
    avgAmp = average |A_m| from time to time + Δt;
    avgPwr = average P_2Δf from time − diff to
    time − diff + Δt;
    if (lastAvgAmp != 0)
        index = fix (ln (avgAmp/lastAvgAmp)/ln β)
        if (index * (avgPwr − lastAvgPwr) < 0)
            score++;
        else score = score−|index|;
    lastAvgAmp = avgAmp;
    lastAvgPwr = avgPwr;
if (score <= 0) an attacker is assumed
```

signalSync() is the function that synchronizes the RF and acoustic signals. Because the speed of sound is much slower than that of radio wave, they do not reach the secondary user simultaneously. In this function, we make use of pauses in human voice. When a pause occurs, the amplitude of sound wave will suddenly drop below 5 in PCM coding (8 bit sampling, max amplitude is 128), and for RF signal, all the power will concentrate at the central frequency (max $P_{2\Delta f}$ is achieved). Two signals are synchronized by sensing these sudden changes. *signalSync* returns the latency of the acoustic signal. More details are presented in Section 7.3.

fix() is to round toward zero, and Δt is set to 80 ms, which is restricted by our experiment equipment. n and β are adjustable parameters. n is the number of the testing rounds. The larger n, the more accurate our method is, but the longer the detection takes. β determines the algorithm's sensitivity of amplitude fluctuation. We have conducted comprehensive experiments with different parameter settings in Section 6.

In the algorithm, only dramatic changes (greater than β times) of A_m take effect, in order to tolerate the fluctuation of f_m. *score* is an evaluation of the relationship between A_m and $P_{2\Delta f}$. If they vary following the lemma in Section 4.1, one point is gained; otherwise, a penalty is made. If the lemma is not followed when the A_m change is extremely large (i.e., $|\text{index}| > 1$), the penalty is heavier correspondingly. At the end of the algorithm, if *score* is equal or less than zero, an emulation attack is reported. The complexity of this algorithm is linear; it spends most of its time waiting for sampling and *signalSync*.

In the environments of a single acoustic source, this algorithm works very well. The evaluations in Section 6 demonstrate its high accuracy and relatively short detection time.

However, the original sound $(\mathrm{w}(t))$ does not always equal to the sound collected by the sound sensor of a secondary user $(\mathrm{w}''(t))$, especially when there are other acoustic sources nearby. While the theorem in Section 4.1 states the relationship between $\mathrm{S}(t)$ and $\mathrm{w}(t)$, this algorithm testifies the relationship between $\mathrm{S}(t)$ and $\mathrm{w}''(t)$. The basis of this algorithm lies in the proximity of $\mathrm{w}(t)$ and $\mathrm{w}''(t)$. We will address this problem in the next section.

5 Noise Mitigation

As mentioned in Section 3, the receiver end of a wireless microphone is typically connected to a loudspeaker or a power amplifier. As a result, $\mathrm{w}'(t)$ (see Fig. 1) should be the main acoustic source in the operating range of a wireless microphone; other acoustic sources are very likely to be weaker than the loudspeaker. In addition, we assume that $\mathrm{w}'(t)$ is very close to $\mathrm{w}(t)$, which is naturally promised by wireless microphone's function.

Based on such assumption, in this section, we will exploit how to utilize collaborative sensing to mitigate the ambient noise and improve the performance of our method for real-world use.

5.1 Mitigating Ambient Noise

Although using microphone arrays to enhance voice input or reduce noise is well researched [15], [16], these approaches cannot be applied to our scenario directly for the following reasons.

First, secondary users do not know each other's location. Classical algorithms assume all the microphones are aligned in an array and their locations are fixed. Second, secondary users are wirelessly connected, so their link bandwidth is limited. Large amount of data exchange is not suitable. Third, traditional microphone arrays are connected to a computer. However, secondary nodes typically have limited computation power. The costs of traditional algorithms are too high for them. In addition, what we want is a distributed solution instead of centralized.

Therefore, we propose a simple but efficient approach as follows: First, the sampling data of acoustic signals are compressed to 11.025 kHz (default sampling rate of most sound cards or sound sensors is 44.1 kHz). Four consecutive samples are averaged into one to reduce communication overhead. We denote compressed sampling data from secondary users A and B by $[a'_1, a'_2, \ldots, a'_n]$ and $[b'_1, b'_2, \ldots, b'_n]$, respectively, and each element is regulated to an 8-bit signed integer. Then, secondary user A requests acoustic samples from B, in response to which B sends $[b'_1, b'_2, \ldots, b'_n]$ to A (synchronization of A and B's acoustic samples is presented in Section 5.2; here, we assume a'_j and b'_j are synchronized). A calculates the cleaned samples a''_j according to the following rule:

$$a''_j = \begin{cases} \dfrac{a'_j + b'_j}{2} & \text{if } (a'_j - a'_{j-1})(b'_j - b'_{j-1}) < 0 \\ |a'_j| > |b'_j| \ ? \ a'_j : \ b'_j & o.w. \end{cases} \quad (6)$$

The idea is to enhance the signal where A and B have the same trend, and smooth the part where they have different opinions. As long as the major components of A and B's samples are the same, this method can alleviate the influence of ambient noise. Evaluation results will be presented in Section 6.

Noise mitigation is executed on a request basis. B sending its samples to A does not necessarily means A also sends to B. The communication overhead is approximately 10 kBytes per second (for one collaborator), which is relatively low. Then, the cleaned samples $[a''_1, a''_2, \ldots, a''_n]$ are used as the input of the Algorithm 1 to calculate *avgAmp*.

The collaboration can be extended to three or more secondary users. It works in an accumulative way, where A calculates with B, and their result calculates with C, for example. The communication overhead becomes nonnegligible when there are a large number of secondary users. The cumulative method saves a large amount of bandwidth compared with exchanging samples between every pair of nodes.

We present the complete protocol in the next section by incorporating noise mitigation into Algorithm 1.

5.2 Emulation Attack Detection with Noise Mitigation

We briefly describe the complete version of our emulation attack detection method as follows:

1. When a secondary user detects a wireless microphone signal, it searches for other secondary users

nearby, and sends each a help request, containing the central frequency where the wireless microphone is found, a time stamp and its own ID.

If a secondary user receives a help request, it replies with a time stamp if it is not busy.

2. The secondary user waits for the reply until timeout. If a neighbor replies, the secondary user extracts the time stamp from the reply and uses it for coarse-grained synchronization. A confirmation is also sent back to the helper to tell it when to start *signalSync* (refer to Section 4.2). This process is repeated if more than one reply is received.
3. The secondary user switches its center frequency to the band where the wireless microphone signal is found, shrinks its band-pass filter to $2\Delta f$, and executes *signalSync* at the same time as it tells helpers.

   On the other hand, upon receiving the confirmation, the helper also adjusts its frequency and bandwidth, and executes *signalSync*. During the execution, the helper records the time when it first observes an RF pause of the wireless microphone signal, and send this time stamp to the help requester.
4. The secondary user checks the time stamp that the helper sends. If the pause the helper detects is the same as its own, an acknowledgment is sent. Then, the helper sends the compressed acoustic samples periodically to the requester, starting from (RF pause time + helper's *diff*). If not, an error message is sent and they both go back to execute *signalSync* again.
5. The secondary user calculates the cleaned acoustic samples according to (6), and uses the results as the input of Algorithm 1. It completes the rest of the task following Algorithm 1.

Noise mitigation requires fine-grained synchronization of acoustic samples between secondary users. The above protocol employs a two-phase approach to achieve this. For the first phase, two nodes exchange their time stamps to perform coarse-grained synchronization. In the second phase, they judge whether the pauses they detected are the same one based on the coarse synchronization, and then use the pause for fine-grained synchronization. As long as they can achieve the accuracy of, for example, 0.1 second in the first phase (it is realistic to expect a reply within 0.1 second for single-hop communication), the fine-grained synchronization can succeed because it is not likely a speaker will pause more than once within 0.1 second.

As mentioned, the noise mitigation process is performed on a request basis. When such a request is not proposed or the collaboration fails, our method regresses to the version without noise mitigation. If a large number of secondary users are colocated, the communication overhead of noise mitigation increases and, as a result, collaboration between some pairs may fail due to interference or limited bandwidth. However, a secondary user does not need to collaborate with every other primary user; collaboration with one or two secondary users is sufficient to achieve large performance improvement (as shown in Section 6).

An alternative way for collaborative sensing is that secondary users sense independently and vote with equal weight afterwards. We will compare this approach with our method in Section 6.

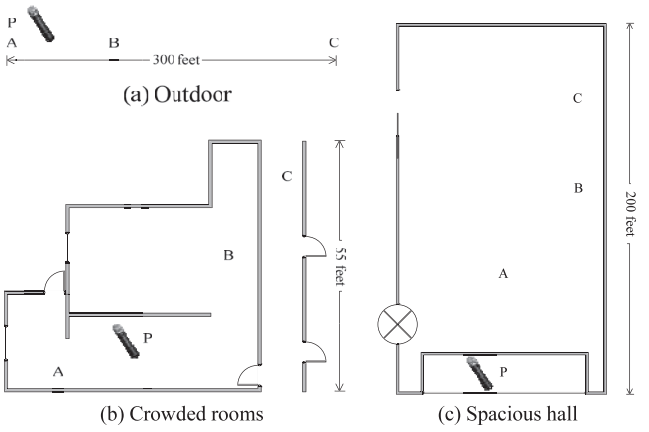


Fig. 4. Layout of experiment environment.

6 Evaluations

We conduct comprehensive experiments to evaluate our method which detects wireless microphone emulation attacks. We test false positives and false negatives with various parameter settings. Both stationary and mobile primary signals are considered. The collaborative sensing (noise mitigating) method is included in the experiments in noisy environments. We also test the time cost of synchronization and the signal attenuation for completeness.

6.1 Experiment Settings

We use wired microphones connected to laptops to collect environmental sound (acting as sound sensors). The raw data collected from sound card are 44.1 kHz and PCM coded. Each sample is an 8-bit unsigned integer. We transform the samples into the range of [-128, 127] and average every four consecutive samples (so that it becomes 11.025 kHz, 11.025 kB/sec) as the input of our algorithm.

We utilize an Agilent E4405B spectrum analyzer to measure the power of RF signals. The minimum sweep time is 80 ms in our settings, which puts constraint on Δt. In all following experiments, Δt is fixed on 80 ms. The detection time will be further discussed in Section 7.2.

Two wireless microphones are used in our experiments. One is working on VHF band (171.9 MHz) and the other on UHF (629.5 MHz). Both have a bandwidth of 200 kHz and 10 mW power output. Their receiver ends are connected to a pair of ordinary loudspeakers (80 watt). Two tape recorders are acting as the source of ambient noise for reproducibility.

We conduct both indoor and outdoor experiments. For the indoor case, we test crowded rooms and a spacious hall, respectively (layouts shown in Fig. 4). The loudspeaker connected to the primary user is put at P. Secondary users are located in A, B, and C, respectively, and their experiment results are averaged unless otherwise specified.

6.2 False Positive and False Negative

We first evaluate the false positive of our method. False positives refer to the detection results erroneously taken a

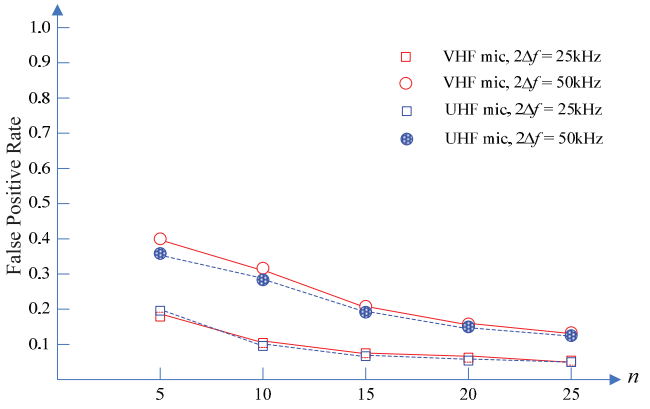


Fig. 5. False positive rate.

genuine primary user as an attacker. In the first experiment, we fix the amplitude sensitivity β to $e(\approx 2.718$, we will vary β later), and test false positives by varying rounds n from 5 to 25 and with different $2\Delta f$ values (50 and 25 kHz). No noise is artificially added; the testing environment is relatively quiet, as well as the following experiments in this section.

The result is shown in Fig. 5. y-axis shows the false positive rate, which equals false positives divided by the total number of tests. x-axis is the number of rounds (n). For each point in the plot, 360 tests are performed (40 at each location in Fig. 4) and various voice samples are tested.

From the figure, we can see that two wireless microphones act very similar. The performance of $2\Delta f = 25$ kHz is much better than that of $2\Delta f = 50$ kHz, which is because too large Δf cannot capture the power variance precisely near the carrier frequency. In the following experiments, $2\Delta f$ equals 25 kHz unless otherwise specified. In both cases, the performance of our method gets better when n becomes larger. When $2\Delta f = 25$ kHz and $n \geq 15$, the false positive rates are less than 0.1, and it is as low as about 0.06 when $n = 25$.

Recall that in our experiments $\Delta t = 80$ ms. That is, 25 rounds take about 2 seconds (can be reduced, see Section 7.2). However, it is not the total detection time of our algorithm, which should include the execution time of *signalSync*. We will test it in Section 6.5.

Fig. 6 shows the false negative rate of our method. False negatives are the cases where an emulation attacker appears but our method fails to report. In this experiment, wireless

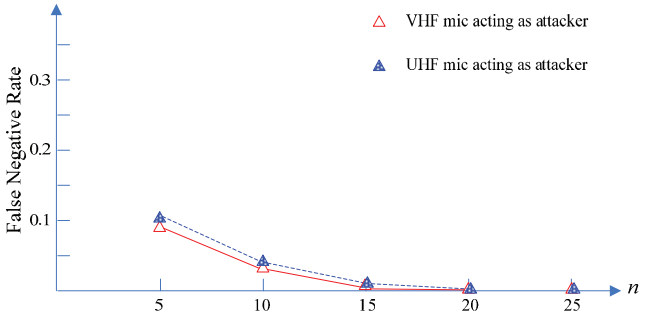


Fig. 6. False negative rate.

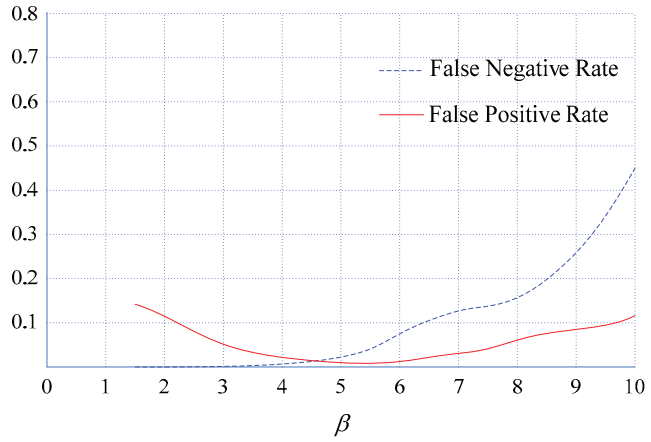


Fig. 7. Impact of β on false alarm and misdetection.

microphones are acting as attackers, with the receiver end disconnected to loudspeakers. We also eliminate $w(t)$ by using line-in as the input of wireless microphones. That is to say, attackers emulate wireless microphones' RF signal perfectly, but without emitting sound (please refer to Section 7.1 to see why emulating sound additionally is not favored or feasible by emulation attackers). Here, only $2\Delta f = 25$ kHz is tested. Other settings are the same as the false positive experiment.

From the figure, we can see the false negative rate of our method is very low. As long as n is larger than 15, there is almost no false negatives. Of course, this test becomes more challenging in a noisy environment, which will be presented in Section 6.4.

As mentioned in Algorithms 1, β determines the sensitivity of the amplitude fluctuation of sound wave. The larger is β, the less is the sensitivity. In the next experiment, we examine the impact of β on the false positive and false negative.

In Fig. 7, we set $n = 25$ and $2\Delta f$ *to* 25 kHz. β varies from 1.5 to 10. The solid line indicates the false positive rate while the dashed line refers to the false negative rate. The results of two wireless microphones are averaged. Other settings are the same as above.

From Fig. 7, we observe that the false negatives increase quickly when β approaches 10. That is because when β is very large, the sensitivity becomes very low and the algorithm can hardly extract any fluctuation of sound amplitude, which causes no penalty and also no awards for *score*. In our algorithm settings, an attacker is not reported when *score* = 0, so that false negatives occur. On the other hand, when β is very small, the false positive rate is relatively high. The reason lies in that high sensitivity makes the algorithm capture even small fluctuations of sound amplitude, which cannot overwhelm the effect of f_m (refer to Section 4). Considering all factors, $\beta \in (3, 6)$ is acceptable.

From the three experiments above, we can conclude that when $n = 25, 2\Delta f = 25$ kHz and $\beta \in (4, 5)$, our method can achieve both false positive rate and false negative rate lower than 0.05 in relatively quiet environments. We set $\beta = 4$ in the following experiments.

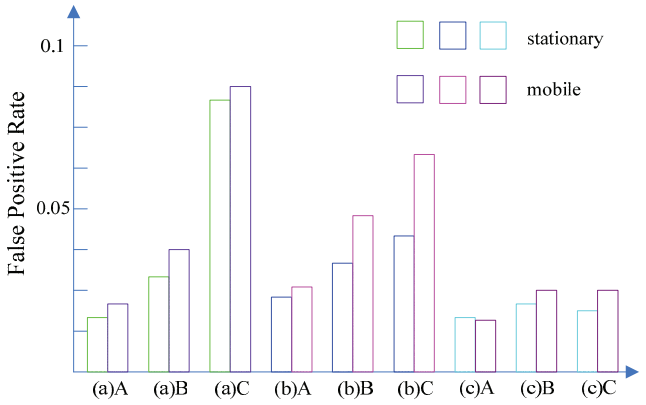


Fig. 8. Impact of mobility and location on performance (1).

6.3 Mobility Test

In this section, a volunteer wears the wireless microphone and moves around at walking speed (approximately 1 meter per second; the loudspeaker do not move). We compare its performance to the stationary case. The tests are conducted in three scenarios (as shown in Fig. 4), respectively. n is set to 25, $2\Delta f = 25$ kHz, and $\beta = 4$.

For the first test, secondary users are stationary. Results are shown in Fig. 8, where (a)A refers to location A in Fig. 4a, and so forth. For each bar, 80 tests are conducted. Generally speaking, mobility does not affect the performance much. All false positive rates are still under 0.1. For the indoor case, the performance in the spacious hall is better than that of the crowded rooms. The influence of mobility is more noticeable in crowded places than open area. We also observe that the performance is more sensitive to the distance between the primary user and secondary user in the outdoor case than indoor, probably because the sound from the loudspeaker is less concentrated and ambient noise is higher for outdoor experiments.

In the next test, both the wireless microphone and the secondary users are mobile. Secondary users also move at walking speed. We do not test the cases of point A, B, and C, respectively, because secondary users now are not stationary. Other settings are the same as above. As we can see from the results (shown in Fig. 9), though the false positive rate is still around or below 0.1, it is slightly higher

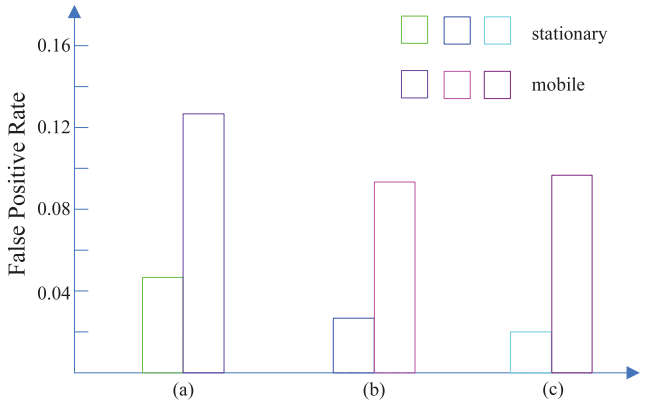


Fig. 9. Impact of mobility and location on performance (2).

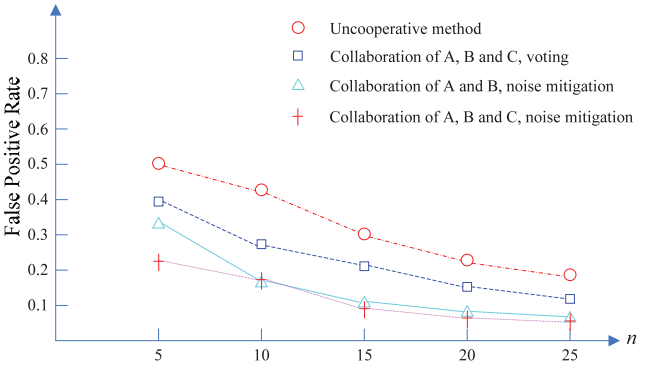


Fig. 10. False positive rates in noisy environment.

than the former test, and the result in spacious room does not outperform the crowded room any more. The reason probably lies in the fact that when both primary and secondary users are mobile, it is more difficult to estimate *diff* (arrival time difference between acoustic signal and RF signal, see Section 4.2). In a spacious room or outdoor environment, users have more space to walk around and the distance between the primary user and secondary users is likely to be longer, which makes the accuracy of *diff* matter more.

We perform the similar tests on false negative rates for both cases as well. The results are all zero or very close to zero (< 0.02), so we do not show the plot.

6.4 Collaborative Sensing

Now, we move on to the evaluation of our noise mitigation approach. In this experiment, we use two tape recorders as noise sources to simulate a noisy environment. Their volume is set to be lower than the loudspeaker connected with the primary user. Three methods are tested and compared. The first is our method without noise mitigation (the same as the experiments above); the second is the one with noise mitigation (see Section 5); and the third one is based on voting.

The latter two are both collaborative methods. The difference is that the third does not exchange acoustic information between secondary users; instead, they detect separately (using the first method) and then exchange the detection results. Majority opinion is adopted. For collaborative methods, secondary users are located at A, B, and C (see Fig. 4), respectively. All acoustic sources ($w(t)$ and noise) use recorded materials for reproducibility.

The experimental results are shown in Fig. 10. Here, $2\Delta f = 25$ kHz and $\beta = 4$. For each point in the plot, 360 tests are performed (40 at each location in Fig. 4). The performance of the original method decreases when the environment is noisy. Even after 25 rounds, the false positive rate is still a little above 0.2. Environmental noise significantly affects the capture of primary users' acoustic signal. The performance of the method with voting is better than the original one. Our method with noise mitigation performs best out of three, in which the false positive rate is around 0.09 when $n = 25$. However, not much difference is observed between two participants and three participants in noise mitigation. Its performance is related to the relative positions between secondary users and noise

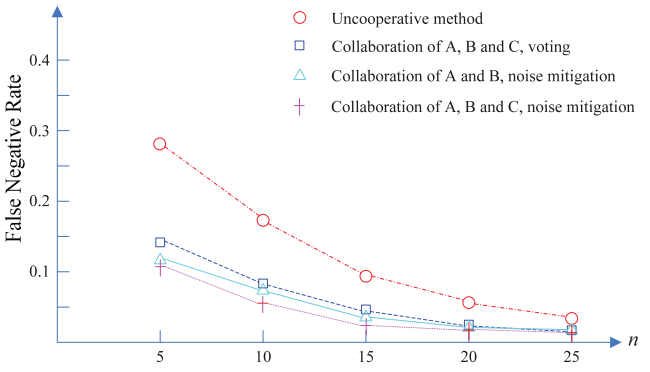


Fig. 11. False negative rates in noisy environment.

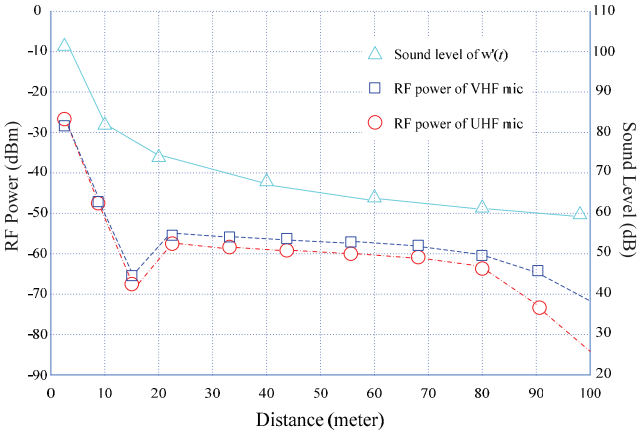


Fig. 12. Attenuation comparison between RF and acoustic signal.

sources. In our settings, noise sources are placed randomly and results are averaged. If all the secondary users happen to be very close to the same noise source, our method may suffer from considerable performance lost.

The results of false negative tests are similar to the false positives (shown in Fig. 11). The difference is that the performance of three methods is closer when n gets larger. Another noticeable result is that the false negative rate of our method is always lower than the false positive rate, no matter the environment is noisy or quiet.

In a nutshell, our method with noise mitigation is able to achieve both false positive and negative rates lower than 0.1 when $n > 20$.

6.5 Synchronization Overhead

In our method, secondary users need to synchronize the detected RF signal and acoustic signal, for that they do not arrive simultaneously. The time of *signalSync* can be divided into two parts: time waiting for the first RF pause and the delay of acoustic signal (*diff*).

Apparently, it is easy to bound *diff*. The operating range of wireless microphones is usually less than 100 meters, and the speed of sound is about 340 meters/sec. Hence, *diff* should be less than 0.3 second. For the other part, we perform experiments to measure the time waiting for the first pause. We test various sound materials, including news reports, lectures, talk shows, etc. The average value is 1.43 seconds per pause. Therefore, the total time of *signalSync* should be less than 1.7 seconds. This is consistent with our measured result in Algorithm 1, which is approximately 1.5 seconds.

6.6 Spatial Attenuation

We have already tested the performance of our method in different locations in Section 6.3. The results show that the performance is slightly better when the secondary user is close to the primary user (or attacker). In this section, the attenuation rates of the RF signal and the sound wave are compared.

The loudspeaker connected with the primary user is turned to medium volume, which is about 102.5 dB measured one meter away. RF power of two wireless microphones are tested (their output power are both 10 mW). The indoor and outdoor results are averaged and shown in Fig. 12. We can see that the power of RF signals have a significant drop at about 15 meters, which is due to multipath and body absorption.

An important observation from this experiment is that when the distance gets close to 100 meters, the power of RF signals quickly drops under -70 dBm, and is not detectable at 110 meters. However, the sound level of $w'(t)$ is still about 60 dB at the same distance, which is more than detectable. This experiment shows that the sound level of $w'(t)$ decreases more slowly than the power of wireless microphone's RF signal. In other words, as long as the RF signal of wireless microphones is detectable, acoustic information, on which our method relies, is always available.

7 Discussion

7.1 Disincentive for Attacker to Mimic Acoustic Signal

In our attack model, we assume that the emulation attacker is able to emulate the RF signal of primary users perfectly, but does not emit sound (see Section 3). Actually, emulating acoustic signal is not feasible or favored by the attackers for the following reasons.

First, if an attacker emulates acoustic signal in order to confuse our method, the sound level should be considerably high. It makes the attacker itself much easier to be found.

Second, the cost of emulation attack becomes higher if attackers emulate both acoustic signal and RF signal. They must be equipped with a loudspeaker and much stronger power supply.

Third, for selfish emulation attackers (see Section 1, one of the two types of primary user emulation attackers), their goal is to occupy the channel and maximize the spectrum usage. If such an attacker mimics acoustic signal, the characteristics of its RF signal should also follow the change of the acoustic signal in order to deceive our method. Hence, it becomes very difficult to further encode any useful information into its RF signal.

7.2 Attack Detection Time

The detection time of our method contains two parts: the execution time of *signalSync* and $(\Delta t * n)$. The first part takes about 1.5 seconds as stated in Section 6.5. From Sections 6.1 and 6.4, we observe that n should be larger than 15 in order to achieve a good performance. Therefore, the detection time of our method is approximately 3 seconds.

Here, we assume Δt equals 80 ms, which is restricted by our hardware (spectrum analyzer).

However, Δt can be largely reduced. The spectrum analyzer we use (Agilent E4405B) scans 400 points within the specified channel to calculate signal power. The way how it works makes Δt considerably large. For a wireless communication device, signal power can be measured much quicker. For example, an 802.11 device can measure RSSI within 1 ms. Therefore, the detection time of our method can be easily reduced to less than 2 seconds.

As mentioned in Section 3, the 802.22 standard draft specifies that wireless microphone signals must be detected within 2 seconds with both false-alarm and misdetection probabilities less than 0.1. Our method is able to achieve this requirement with proprietary wireless devices.

7.3 Signal Synchronization

In Section 4.2, we briefly explained that *signalSync* utilizes pauses in human voice to synchronize the acoustic signal with RF signal. Here, we describe more details.

When a pause in the RF signal is detected, in the next 0.3 second, *signalSync* waits for the first acoustic pause. If it comes, synchronization succeeds. We choose 0.3 second because the RF coverage of wireless microphone is usually less than 100 meters and 0.3 second is sufficient for sound wave to travel that far. We only search for the first acoustic pause after the RF pause because it is unlikely that a speaker will pause more than once within 0.3 second. If no acoustic pause is detected when 0.3 second elapses, *signalSync* restarts from beginning. It reports an attacker after three restarts. Although it uses a number of heuristics, this function works well in practice.

There are some existing methods for signal delay estimation, such as cross-power spectrum phase and LMS filtering [17]. This topic is orthogonal to our work, and we can make use any of them for signal synchronization. We use the method described above because of its simplicity and low computation overhead.

8 Conclusion

In this paper, we proposed a novel method to detect emulation attacks of mobile primary users (wireless microphones). The relationship between the RF signal and acoustic signal are exploited to differentiate attackers from genuine wireless microphones. We also developed a noise mitigation method to improve the detection accuracy in noisy environments.

We conducted experiments to evaluate our method. The results demonstrate that our method can achieve both false positive rate and false negative rate lower than 0.1 within 3 seconds even in a noisy environment. The detection time can be further reduced when proprietary white-space devices are available. Finally, PUE attack detection is one side of the problem; the study of anti-PUE communication schemes is also worth further attention.

Acknowledgments

This research was supported in part by the US National Science Foundation (NSF) through grant CNS-0831914 and the US Army Research Office through MURI grant W911NF-07-1-0318.

References

[1] R. Chen and J.M. Park, "Ensuring Trustworthy Spectrum Sensing in Cognitive Radio Networks," *Proc. IEEE Workshop Networking Technology for Software Defined Radio Networks,* pp. 110-119, Sept. 2006.

[2] Federal Communications Commission, "Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Spectrum Agile Radio Technologies," ET Docket Nos. 03-108, Dec. 2003.

[3] R. Chen, J.M. Park, and J.H. Reed, "Defense against Primary User Emulation Attacks in Cognitive Radio Networks," *IEEE J. Selected Areas in Comm.,* vol. 26, no. 1, pp. 25-37, Jan. 2008.

[4] R. Chen, J.M. Park, and K. Bian, "Robust Distributed Spectrum Sensing in Cognitive Radio Networks," *Proc. IEEE INFOCOM,* pp. 1876-1884, Apr. 2008.

[5] Z. Jin, S. Anand, and K.P. Subbalakshmi, "Detecting Primary User Emulation Attacks in Dynamic Spectrum Access Networks," *Proc. IEEE Int'l Conf. Comm. (ICC),* June 2009.

[6] Z. Jin, S. Anand, and K.P. Subbalakshmi, "Mitigating Primary User Emulation Attacks in Dynamic Spectrum Access Networks Using Hypothesis Testing," *Proc. ACM Mobile Computing and Comm. Rev.,* 2009.

[7] *IEEE Standard 802.22-09/0068r1, Sensing Performance with the 802.22.1 Wireless Microphone Beacon,* IEEE, Mar. 2009.

[8] Z. Lei and F. Chin, "A Reliable and Power Efficient Beacon Structure for Cognitive Radio Systems," *Proc. IEEE Int'l Conf. Comm. (ICC),* May 2008.

[9] I.F. Akyildiz, W. Lee, M.C. Vuran, and S. Mohanty, "Next Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Network: A Survey," *Computer Network,* vol. 50, pp. 2127-2159, 2006.

[10] D. Cabric, S.M. Mishra, and R.W. Brodersen, "Implementation Issues in Spectrum Sensing for Cognitive Radios," *Proc. Asilomar Conf. Signals, Systems and Computers,* vol. 1, pp. 772-776, Nov. 2004.

[11] A. Ghasemi and E.S. Sousa, "Collaborative Spectrum Sensing for Opportunistic Access in Fading Environments," *Proc. IEEE Int'l Symp. New Frontiers in Dynamic Spectrum Access Networks (DySPAN),* Nov. 2005.

[12] A.W. Min, X. Zhang, and K.G. Shin, "Spatio-Temporal Fusion for Small-Scale Primary Detection in Cognitive Radio Networks," *Proc. IEEE INFOCOM,* Mar. 2010.

[13] E. Reihl, "Wireless Microphone Characteristics," IEEE 802.22-06/0070r0, May 2006.

[14] M.J. Ryan and M. Frater, *Communications and Information Systems.* Argos, 2002.

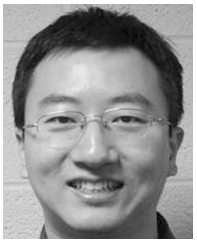
[15] D. Giuliani, M. Omologo, and P. Svaizer, "Experiments of Speech Recognition in a Noisy and Reverberant Environment Using a Microphone Array and HMM Adaptation," *Proc. Int'l Conf. Spoken Language (ICSLP),* pp. 1329-1332, Oct. 1996.

[16] T. Yamada, S. Nakamura, and K. Shikano, "Simultaneous Recognition of Multiple Sound Sources Based on 3-D n-Best Search Using Microphone Array," *Proc. Eurospeech Conf.,* vol. 1, pp. 69-72, Sept. 1999.

[17] M. Omologo and P. Svaizer, "Acoustic Source Location in Noisy and Reverberant Environment Using CSP Analysis," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP),* pp. 921-924, 1996.

[18] S. Chen, K. Zeng, and P. Mohapatra, "Hearing Is Believing: Detecting Mobile Primary User Emulation Attack in White Space," *Proc. IEEE INFOCOM,* 2011.

Shaxun Chen received the BSc and MSc degrees in computer science from Nanjing University, China, in 2005 and 2008, respectively. He is currently working toward the PhD degree in the Department of Computer Science at the University of California, Davis. He was a visiting researcher in the Institute for Infocomm Research, Singapore, in 2007. His research interests include wireless security, network forensics, and video forensics.



Kai Zeng received the bachelor's degree in communication engineering and the master's degree in communication and information system from the Huazhong University of Science and Technology, China, in 2001 and 2004, respectively. He received the doctoral degree in electrical and computer engineering at Worcester Polytechnic Institute (WPI) in 2008. He is currently an assistant professor in the Department of Computer and Information Science at the University of Michigan-Dearborn. Before that, he was a postdoctoral scholar in the Department of Computer Science at the University of California, Davis (UCD). He won the Sigma Xi Outstanding PhD Dissertation Award at WPI in 2008 and the Excellence in Postdoctoral Research Award at UCD in 2011. His current research interests are in wireless security and networking with emphasis on physical layer security, network forensics, and cognitive radio networks.

Prasant Mohapatra received the doctoral degree from Penn State University in 1993. He is currently the Tim Bucher family endowed chair professor and the chairman of the Department of Computer Science at the University of California, Davis. In the past, he has been on the faculty at Iowa State University and Michigan State University. He has also held visiting scientist positions at Intel Corporation, Panasonic Technologies, Institute of Infocomm Research, Singapore, and National ICT Australia. He has been a visiting professor at the University of Padova, Italy, and Yonsei University, South Korea. He was/is on the editorial boards of the *IEEE Transactions on Computers*, *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Parallel and Distributed Systems*, *ACM WINET*, and *Ad Hoc Networks*. He has been a guest editor for *IEEE Network*, *IEEE Transactions on Mobile Computing*, *IEEE Communications*, *IEEE Wireless Communications*, and the *IEEE Computer*. His research interests are in the areas of wireless networks, sensor networks, Internet protocols, and QoS. He received the Outstanding Engineering Alumni Award in 2008. He is a fellow of the IEEE.