2021 IEEE Conference on Communications and Network Security (CNS)

CompKey: Exploiting Computer's Electromagnetic Radiation for Secret Key Generation

Fangfang Yang
UC Riverside

Mohammad A. Islam
University of Texas at Arlington

Fan Wu
Shanghai Jiao Tong University

Shaolei Ren
UC Riverside

***Abstract*—Secure communication between wireless devices has become an important issue for mobile devices. To achieve this, many efforts have been devoted to leveraging random ambient signals to generate a shared secret key between participating devices. However, the existing methods often suffer from limited authentication distance (e.g., less than 4cm for WiFi-based secret key generation). In this paper, we propose a novel secret key generation method, called CompKey, which allows wireless devices in the proximity of a computer to securely associate with each another by exploiting electromagnetic radiation (EMR) emitted from the computer. Unlike the existing approaches that often rely on ambient signals' amplitudes, CompKey extracts the EMR's frequency information, which fluctuates randomly with time, as a secret source. We design a difference-based encoding method, which preserves randomness of the frequency and avoids many consecutive zeros and ones in the generated key. We show via experiment evaluation that participating devices can reliably achieve around 10 bits/s bit generation rate and 100% key matching rate when they are located within 50cm away from the source computer. Moreover, the experiment results with the presence of attackers demonstrate that our method is robust against eavesdroppers and strong copy attackers who can imitate the key generation process.**

I. Introduction

Device-to-device (D2D) technology is a crucial part of the next-generation wireless communications, allowing mobile devices to communicate with each other directly when they come into proximity [1], [2]. Nonetheless, it also faces privacy and security concern because of the broadcast nature of wireless transmission [3], [4], [5], [6], [7].

Traditionally, to guarantee security and secrecy, the transmitted data is encrypted through cryptographic techniques, such as the classic public key encryption Diffie-Hellman (DH). However, since DH does not verify the identity of participating devices, an attacker with a directional antenna could easily impersonate a legitimate device and establish a shared key with one or both of the valid devices [8], [9], [10], [4].

To address the limitations of DH, recent researchers exploit the dynamic characteristics of natural ambient signals in order to construct a secure and authentic communication channel between two or more collocated devices. Concretely, devices in proximity of each other can derive a shared secret key from their common time-varying ambient environment. In addition, the physical proximity can serve as proof of device authentication. The ambient sources appearing in literature include radio related signal, such as RSSI and CSI [6], [11], [5], [12], [4], [13], ambient audio signal, ambient luminosity [14], [15], [16], [17] and biometrics [18], [7], [19].

However, the existing methods have significant limitations. First, these methods may require legitimate devices to be placed very close to each other, e.g., less than 4cm in WiFi-based approaches [6], [4], [5]. The reason is that they use signals' amplitude attribute for key generation, which changes dramatically beyond a half-wave length distance. Similarly, to extract identical biological characteristics for secret key generation, participating devices have to be touched by a single person [18], [7], [19]. Second, some of these methods rely on specialized sensing apparatus. For example, leveraging biological signals requires specialized receiving sensors, like electromyography sensor or accelerometer [18], [7]. Third, some of these methods may take a long time to generate the secret key. For example, techniques based on Received Signal Strength Indicator (RSSI) have largely limited key generation rates, since only one RSSI value can be extracted from one packet [6], [11], [5], [13]. Exploiting the ambient sound or ambient luminosity based on its (slow-varying) statistics is also subject to a low key generation rate [17], [20]. Last but not least, some of these methods require two devices directly exchange signals between each other (e.g., channel reciprocity), which makes it not suitable for multiple-device key generation [9], [21].

In this paper, we discover a computer's electromagnetic radiation (EMR) signal as a **localized** (hence secure only for devices nearby the computer), and **randomly varying** ambient signal for secret key generation. Moreover, we exploit the computer EMR's **frequency** information, instead of amplitude (like WiFi-based approaches [9]), to overcome the half-wavelength distance requirement and preserve integrity among legitimate devices that are within a range of the source computer (empirically 50cm in Section VI).

Concretely, we design CompKey, a scheme leveraging EMR over the memory bus clock frequency of a computer to generate shared secret key between two or more legitimate devices which are close to the computer. In order to extract the frequency information of the radiation, we first perform Fast Fourier Transform (FFT) to get the frequency spectrum and locate the frequency band based on the frequency of the most dominant spike. We then filter the signal over this frequency band to get rid of other interfering spikes. Based on the filtered signal, we extract the frequency of the highest spike at every time step and get the time-varying frequency information. This paper adopts difference-based encoding method instead of the popular quantization approach, which is prone to long

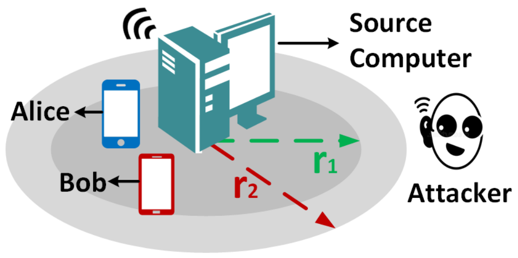


Fig. 1. Two parties (Alice and Bob), who are located in the vicinity of a computer (within r_1 from the computer), can perceive the same radiation emitted from the computer. An attacker can only reside beyond some distance, r_2, from the source computer.

run zeros and ones. We convert the frequency difference of two adjacent time steps to binary bits and use reconciliation process to get rid of the bit discrepancy caused by fading effect and imperfection of measuring equipments. To evaluate CompKey, we conduct experiments in our lab office, showing that CompKey is a reasonably fast and robust secret key generator, which can achieve 10 bits/s bit generation rate and 100% key matching rate.

II. Problem and Threat Model

A. Problem Definition

Fig. 1 provides an illustration of CompKey. Alice and Bob are located within r_1 from a computer (hereafter called source computer) and would like to securely communicate with each other. An attacker can only reside r_2 away from the source computer to launch attacks. In practice, r_1 and r_2 can be both conservatively chosen (i.e., small r_1 and large r_2) to ensure that legitimate devices can have a high success rate of key generation while keeping attackers away at a safer distance.

Source Computer. A computer that emits EMR signals for nearby legitimate devices for authentication is called a source computer. The source computer can be a work desktop or regular personal laptop, but cannot be the small size tablet computer that does not emit significant EMR.

Legitimate Devices. These are devices located near the source computer and would like to securely communicate with each other. Note that the source computer itself can also be a legitimate device. In Section VI, we will show that legitimate devices need to be located within 0.5m from the source computer. Legitimate devices can receive the source computer's EMR signals, whose frequency ranges (e.g., around 800MHz and 1600MHz) are close to those of current cellular/WiFi signals. In other words, the existing antenna on mobile devices for receiving cellular and WiFi signals is also capable of capturing computer's EMR signals, provided that its receiving frequency is tuned to proper frequency ranges.

Potential applications. Nowadays, it is very common that devices interact with each other nearby through wireless channels. For example, mobile devices are often used for verification purposes to access laptop or other office resources, while personal health data is frequently exchanged between wearable devices and smart phones. By using CompKey, we can build a local circle of trust for interaction between nearby devices around a source computer.

B. Threat Model

We assume that legitimate devices are located in the proximity of a source computer and malicious adversaries can only launch attacks at a distance of at least r_2 away from the source computer. The attacker cannot put any tapping device around the source computer as otherwise it would be discovered when approaching the computer. We consider the following two kinds of attacks.

Eavesdropping attack. The attacker can overhear the EMR signal emitted from the source computer at a distance and also eavesdrop the legitimate devices' communication.

Copy attack. The attacker can not only capture all the information transmitted over public wireless channel but also obtain component details of the source computer. Moreover, the attacker is able to imitate the computer's working status during the key generation process. We refer to this strong adversary as copy attacker. Thus, the copy attacker is able to find another computer with exactly identical memory bus clock and play the same programs in that computer to imitate the memory bus status. The attacker then records the radiation wave and generates its own key following CompKey steps.

C. Remarks

- **Software-defined radio (SDR) capability.** While the size of existing antennas on mobile devices is suitable for capturing computer's EMR signals, the receiving frequency needs to be tuned to proper frequency ranges. Fortunately, such SDR capability is being integrated by major vendors like Intel into baseband solutions (to accommodate multi-standard communications with a low cost) [22]. Thus, SDR does not present an insurmountable barrier and CompKey will be more universally applicable in future devices.
- **Synchronization.** Like in other proximity-based authentication schemes [6], [17], [9], legitimate devices need to be synchronized for secret key generation by using CompKey. The synchronization requirement in CompKey is easy. Specifically, as shown in our experiments in Section VI, CompKey extracts EMR frequency every 0.08s, for which synchronization requirement is much less stringent than for normal communication that needs millisecond-level synchronization.
- **Authentication distance.** Through empirical evaluation, we consider $r_1 = 0.5$m away from the source computer as the authentication distance, within which devices can reliably extract secret keys. On the other hand, attackers are kept at a distance of $r_2 = 2$m away from the source computer. Although 1.5m is already enough to stop attackers from getting keys, we use 2m as a *safer* distance limit. This is common in the literature of proximity-based authentication [6], [5] where a distance greater than the authentication distance is assumed to keep attackers away.
- **Other attacks.** We discuss a few other attacks.

Jamming attacks. In our threat model, the attacker's goal is to obtain secret keys, instead of completely blocking Bob/Alice communications. If very strong noise is injected, then CompKey may not work, but the attacker cannot obtain keys either. Thus, like in the existing proximity-based authentication [6], [7], [19], [20], [5], [15], we do not consider attackers who inject or jam radio signals. In fact, legitimate devices can also easily discover the existence of such an attacker: in the presence of such an attacker that injects a high EMR signal to mimic the source computer's signal, the resulting EMR signals received by legitimate devices will not decrease significantly when the legitimate devices moves some distance away from the source computer.

Untrustworthy source computer. Like in the existing literature [23], [20], we assume the EMR signal produced by the source computer is not directly compromised by attackers. Even if an attacker can compromise a source computer and acquire its random EMR signals, it also needs to know exactly when Bob and Alice tap into EMR signals to extract keys, which can be non-trivial for attackers.

• **Limitations.** CompKey is designed to provide *additional* protection (e.g., as part of multi-factor authentication). If our threat model is violated (e.g., an attacker is hidden inside a compromised source computer), then CompKey may not work as designed, which is also the limitation in other proximity-based authentication [6], [7], [19], [20], [5], [15]. In such a case, however, it can still be non-trivial to acquire the secret key because the attacker also needs to know exactly when Bob and Alice tap into random EMR signals for key generation.

III. Characteristics of Computer EMR

A. Memory Bus EMR

Random access memory (RAM) refers to computer memory that temporarily stores and retrieves data at a high speed, which will be processed by the CPU. Data transferring process between CPU and RAM is controlled by memory bus clock. During this process, state transition of digital circuit will induce transitioning voltage signal, which causes changes in electric fields. Data transferring through memory buses will introduce an alternating current, which leads to variation in magnetic fields. The combination of changing electric field and changing magnetic field is termed electromagnetic field (EMF). Digital data paths connecting CPU and RAM have long unterminated wires and can serve as antennas, through which EMF can be radiated to the outside world as EMR. Since the EMF changing rate is controlled by memory bus clock, the EMR frequency should correspond to the memory clock frequency [24].

In order to validate this, we collect the radiation signal from four different computers, the detailed information of which is provided in Table I. For each computer, we collect the radiation in situations when the computer is turned on and off. We use a Universal Software Radio Peripheral (USRP) to capture the emitted radiation, tuning it to 20MHz frequency range centered at the the computer's referred memory bus clock frequency. After that, we perform Fast Fourier Transform (FFT) to get its frequency spectrum with 1Hz frequency resolution, which is shown in Fig. 2. We can see that for each computer compared to the case when the computer is turned off, there are prominent spikes appearing and clustering around the its memory bus clock frequency when it is turned on. This further verifies that the memory bus clock frequency information can be disclosed through the radiation.

TABLE I
Memory bus clock frequency information of different computers.

Computer	CPU	RAM	MC (MHz)
HP ProBook 450	Core i5-6200U	DDR4 SDRAM	1600
Dell XPS 8920	Core i7-7700	DDR4 SDRAM	1200
Acer Aspire V3-372T	Core i5-6200U	DDR3L SDRAM	800
Dell OptiPlex 9020	Core i5-6200U	DDR3 SDRAM	800

Note: MC represents memory clock frequency.

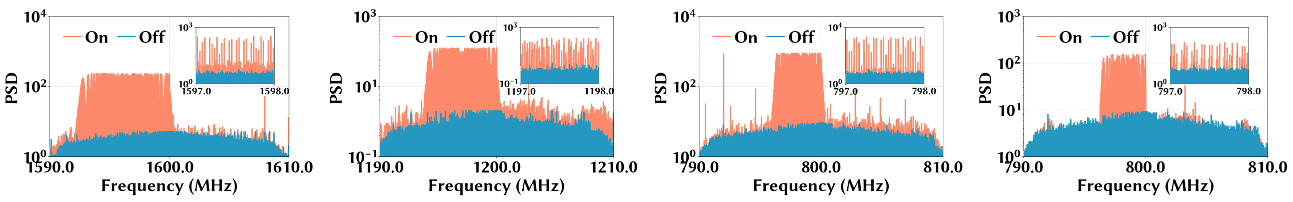
B. Memory Bus EMR as a Secret Source

The EMR signal needs to meet the following requirements.

1) Temporal Variation: Memory bus clock is controlled by a clock generator, an electronic oscillator, which aims to synchronize the data transferred between CPU and RAM. Due to minor variations in temperature, silicon characteristics and local electrical conditions, these crystal-based oscillators are subject to diverge and run at slightly different rate from the reference frequency, which is termed clock drift. This physical phenomenon is proved genuinely random and acts as the non-deterministic random source in many hardware random-number generators [25]. Since the frequency of emitted EMR derives from memory bus clock, the frequency information of the EMR signal is accordant with clock drift, which makes it a good candidate to be a random source in authentication key generation.

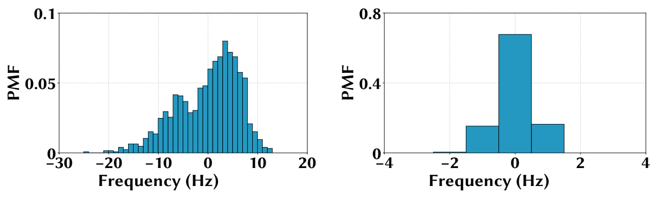
We run empirical experiments to validate the randomness of the desired EMR frequency. Using USRP, we collect the radiation for 100 seconds. Over every 0.1s, we perform FFT and extract the frequency information. We get 1000 frequency samples in total. We will explain how to track the EMR frequency in Section IV. For brevity, we only show results for the Acer Aspire V3 computer. Fig. 3(a) gives the probability mass function (PMF) of the 1000 frequency samples. Note that the frequency value shown in the figure is shifted to have a zero mean. We can see that the frequency distributes randomly over 50Hz frequency range. We extract the frequency information of EMR signals, instead of their amplitudes, because amplitudes of EMR signals change dramatically with the distance and remain approximately unchanged within less than a half wavelength. In other words, if we use EMR's amplitudes as the randomness source for key generation, our authentication distance would be less than 10cm given the EMR's wavelength.

2) Integrity and Authenticity: As long as a device is close to the computer, it will sense the radiation with sufficient energy and extract the EMR spikes through FFT analysis. While EMR signal amplitudes varies significantly over distance, its frequency is less affected by distance, thus meeting the integrity requirement. Fig. 3(b) gives the distribution of frequency difference between two participating users, who are placed near a source computer and synchronously detect the radiation emitted from it. We can see that two users get exactly the same normalized frequency for around 70% of the time. The absolute frequency difference between two users



(a) HP ProBook 450 (b) Dell XPS 8920 (c) Acer Aspire V3-372T (d) Dell OptiPlex 9020

Fig. 2. Frequency spectrum of memory bus EMR for four different kinds of computers. The blue line represents the EMR that is obtained when the computer is off. And the red line is the captured signal when the source computer is turned on.



(a) (b)

Fig. 3. (a) Histogram of EMR frequency of Acer Aspire V3. (b) Histogram of frequency difference between two devices both collecting EMR signals 0.5m away from Acer Aspire V3.

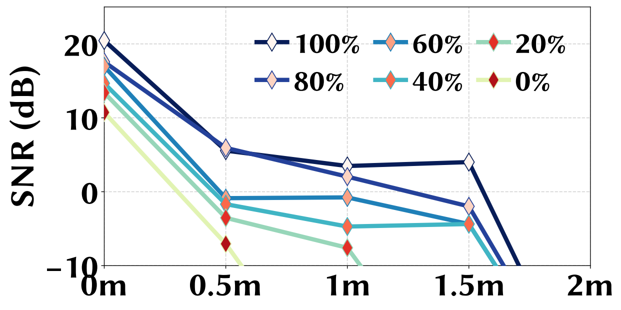


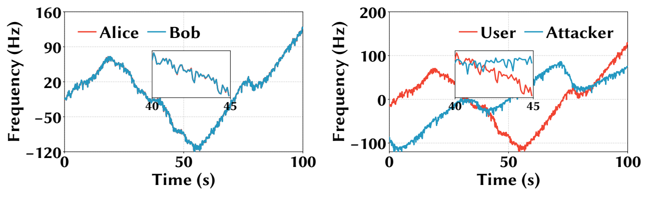
Fig. 4. Acer Aspire V3-372T: SNR vs. distance under different RAM access frequencies.

is less than 2Hz for almost 100% of the time, which further corroborates the integrity requirement.

Like the existing proximity-based authentication [6], [9], authenticity is ensured by not allowing malicious attackers approaching the source computer when executing key generation process. Because of the fading effect over distance, the attacker far away from the source computer will not detect the radiation clearly and barely get the secret key. Thus, the party who could generate the correct secret key by hearing the radiation of the source computer is the one that is in the vicinity of the computer and hence is a legitimate participating device by our threat model.

3) Confidentiality: Confidentiality is another criterion for the signal to be a secret key source. We will elaborate confidentiality of memory bus EMR from the perspective of eavesdropping attacker and copy attacker.

a) ***Eavesdropping Attacker:*** As a computer's EMR is naturally a low-energy signal for human safety and compliance requirement [26], the emitted EMR can not propagate a long distance. In addition, the electromagnetic waves are subject to inverse-square law propagation loss. Thus, as long as the eavesdropper is kept some distance away from the source



(a) (b)

Fig. 5. (a) The normalized EMR frequency variation pattern during 100s when Alice and Bob are placed close to a source computer and synchronously collect the radiation data. (b) The normalized EMR frequency variation pattern of Alice and an attacker. The attacker launches a copy attack and collects EMR using its receiving device.

computer, it can not capture the radiation with sufficient energy and is not able to get accurate frequency information.

To show the fading effect over distance, we show the experimental results for Acer Aspire V3-372T, while the results for other computers are similar. We control and change RAM access frequency from 0% to 100% (i.e., percentage of time the RAM is transferring data with CPU). For each RAM access frequency, we collect EMR signals from different distances, from 0m to 2m, and calculate signal to noise ratio (SNR) of the radiation. The value of SNR in dB can be calculated as $SNR = 10 \times \log_{10}(P_s/P_n)$, where P_s is power of EMR signal (excluding noise power) and P_n is power of noise. Fig. 4 demonstrates the SNR results, showing that EMR becomes very weak beyond 1.5 meters. In our later evaluation section, we can see that for accurate authentication, devices need to be placed within 0.5 meters away from the source computer, while an attacker located 2m away from the source computer can barely receive the EMR signal for secret key extraction (as further validated in Section VI).

b) ***Copy Attacker:*** In order to verify the robustness of CompKey against copy attackers, we set up a copy adversarial scenario. Taking computer Acer Aspire V3 as an example, which is running a Python program, we capture the EMR signals through two USRP antennas, which act as Alice and Bob and are placed near the source computer. We collect EMR signals for 100 seconds and extract one EMR frequency every 0.1s. The frequency variation pattern of Alice and Bob is shown in Fig. 5(a). To simulate the copy attacker, we use the same computer, play the same Python program and capture the emission for another 100 seconds serving as copy attacks. Fig. 5(b) shows the normalized frequency variation pattern of Alice and Attacker. We can observe that Alice and Bob extract nearly identical EMR frequency, with a correlation coefficient of 0.99. On the other hand, even if the sophisticated copy attacker obtains the source computer and imitates the source computer's RAM activity, it cannot obtain the same frequency variation pattern. From Fig. 5(b) we can see that two variation patterns are dramatically different from each other, with only 0.08 correlation coefficient.

To sum up, the above observations confirm that the frequency of a computer's EMR signal is **localized** and **random**, and provide a strong support for exploiting a computer's EMR to generate shared secret keys for nearby devices.

IV. The Design of CompKey

A. Extraction of EMR Frequency Signal

In the previous section, we know that legitimate devices close to a source computer can extract the EMR frequency of the source computer. From Fig. 2, we can see that the receiving devices will get a spike cluster in frequency spectrum of the captured EMR signal. The frequency of any one of these spikes will vary in accordance with the memory clock frequency. To validate this, we calculate the correlation coefficient of frequency variation between each two spikes and show that all these spikes have the same variation trend, which means that the frequency of any of these spikes can represent the clock frequency.

We notice that in the spike cluster each two adjacent spikes are separated by around 30kHz, whereas the varying range of the memory clock frequency is comparably small (less than 1kHz). Thus, once we locate one dominant spike in the cluster, the frequency of which is denoted as λ, we can use a bandpass filter, with frequency band $[\lambda - \Delta\lambda, \lambda + \Delta\lambda]$, to preserve this particular spike and get rid of other spikes which all vary in the same manner. For example, we can easily get the frequency variation pattern and track frequency of the highest spike of the filtered signal.

B. Difference-based Encoding Method

A straightforward approach is utilizing a quantization approach, which divides the selected frequency value into several levels and encodes the level into binary bits. Nonetheless, it is subject to a large number of consecutive ones or zeros [27], because the EMR frequency only changes marginally over time. Considering this, we choose to convert the frequency difference between two adjacent time steps into binary secret bits. This decision is based on the following two observations. First, the frequency variation is highly random (albeit small) and two participating parties, observing the same varying source, obtain similar patterns, which can be seen from Fig. 5(a). Second, the attacker, even if using the same computer and playing the same program, gets significantly different variation patterns from legitimate ones, as shown in Fig. 5(b).

The complete encoding process is elaborated in Algorithm 1. The first step is to get EMR frequency of every time step. In this stage, we first have participating devices collect EMR signals synchronously for a period of time over the frequency band centered at the clock frequency. Then, we divide the EMR signal into non-overlapping segments, each with Δt time window size. After that, we first perform FFT analysis over the first segment, get the frequency of the highest spike, denoted as λ, and then filter the signal using a bandpass filter with a passband, $[\lambda - \Delta\lambda, \lambda + \Delta\lambda]$. In our experiment, $\Delta\lambda$ equals to 50Hz. Finally, we perform FFT on each segment of the filtered signal and get the frequency of the highest spike. After the first stage, we will obtain a frequency list which includes the EMR frequencies of each time step.

Algorithm 1 Difference-based Secret Key Generation

```
1:  **Input:** Memory bus EMR signal $S$, Sampling rate $f_s$, FFT
    time window size $\Delta t$, Encoding threshold $\sigma$
2:  **Output:** Secret bit list $B$
3:  Divide EMR signal $S$ into $M$ segments each with $\Delta t$ size.
4:  Perform FFT to the first segment of EMR signal.
5:  Get the frequency of the highest spike of the first segment,
    denoted as $\lambda$.
6:  Initialize a frequency list $f$ with M elements.
7:  for $i = 1, 2, \cdots, M$ do
8:      Filter the $i$-th segment with passband $[\lambda - \Delta\lambda, \lambda + \Delta\lambda]$.
9:      Perform FFT to the filtered signal.
10:     Get the frequency of the highest spike.
11:     Store the frequency into the $i$-th element of $f$.
12: end for
13: for $j = 1, 2, \cdots, M$ do
14:     Get difference of $(j + 1)$-th and $j$-th frequency in $f$.
15:     Compare the difference with $\sigma$.
16:     Encode the difference according to comparison.
17:     Append the encoded bits to $B$.
18: end for
19: return $B$
```

The second stage of the algorithm is to encode the frequency difference between two adjacent time steps. Here, we introduce an encoding threshold parameter σ. If the frequency value of current time step is larger than the frequency of previous time step by σ, we regard it as rising and encode it as $'11'$. If the current frequency value is σ less than the previous one, it is treated as dropping and encoded as $'00'$. In the remaining cases, the absolute difference between the present frequency and the previous frequency is less than and equal to σ, and hence it is regarded as unchanged and encoded as $'01'$. Traversing the entire duration, each device will get its own secret bits.

C. Reconciliation

Reconciliation is a widely-employed method to mitigate and even eliminate minor mismatching bits between two bit sequences via Error Correction Coding (ECC) [5], [7]. The $C(n, k, r)$ ECC scheme can encode k bits data into a valid n bits codeword by adding $(n - k)$ parity bits, which is a one-to-one encoding function. For a clear representation of reconciliation process, we use $f()$ and $g()$ to represent encoding and decoding functions, respectively. Use k_a and k_b, two n-bit strings, to represent the bit strings obtained by Alice and Bob. First, Alice calculates the corresponding valid codeword closest to her bit string, $f(g(k_a))$. Then, Alice computes an offset, $\delta = k_a \oplus f(g(k_a))$, between her bit string and the codeword. Alice transmits δ to Bob by public medium, which means adversaries can also detect this offset. After receiving the offset, Bob can deduce a bit string by the following equation, which equals k_a with a high probability: $k'_a = \delta \oplus f(g(k_b \oplus \sigma))$. If the mismatching bit between Alice and Bob is no larger than r, k'_a will be equal to k_a. With the reconciliation process, Alice and Bob can ultimately possess an identical secret key with a high likelihood.

D. Privacy Amplification

Theoretically, there are $(n-k)$ bits of the shared key leaked to the attacker through the offset δ sent over the public medium. In order to get rid of the $(n-k)$ bits leakage, Alice and Bob can use the decoded version of k_a as the final authentication key, $g(k_a)$, which is a k-bit string, instead of directly using k_a. This way, however, sacrifices the bit generation rate, reducing it by a factor of $\frac{n-k}{n}$.

V. Experimental Methodology

A. Experiment Setup

Experiment location. All the experiments are conducted in our lab office, which is an open space with more than 30 workstations. Each workstation is equipped with an off-the-shelf desktop computer and each two workstations are separated about 1.5 meters away from each other, as illustrated in Fig. 6. We focus on Dell XPS and Dell OptiPlex as the source computers, respectively, because we have other computers of the same configuration that can act as strong interference to expose CompKey to an undesired environment. In each experiment, there exists an interfering computer 1.5m away from the source computer. Other computers are not shown in Fig. 6 because they are more than 2m away from our source computers and hence have negligible interference.

Experiment prototype. The experiment prototype of CompKey includes a computer as the radiation source and USRP X310 to collect the radiation signals. The USRP X310 is embedded with UBX 160 daughterboards with a LP0965 Log Periodic PCB antenna, acting as participating devices. The collected signal will be transferred to our HP ProBook 450 computer and processed by CompKey, which is implemented in Python 2.7.

Signal collecting and processing. For a specific source computer, the receiving frequency band of USRP will be tuned to 2MHz centered at the computer's reference memory clock frequency. Each participating device synchronously collects EMR signals and slices the collected signals into non-overlap segments, each with 0.08s time window size. FFT will be performed over each segment to get the EMR frequency information.

Encoding threshold. CompKey uses a difference-based encoding algorithm to convert the frequency variation into binary bits based on an encoding threshold σ. We set σ to be 2Hz. Specifically, if the frequency difference between two adjacent times is larger than 2Hz, it will be encoded to '11'.

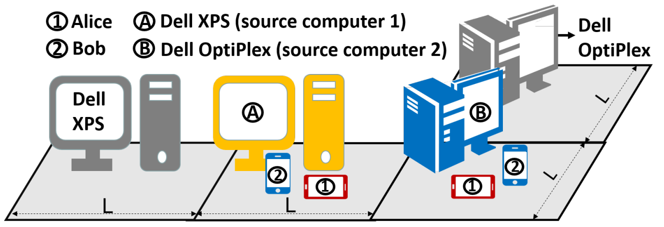

① Alice
② Bob
Ⓐ Dell XPS (source computer 1)
Ⓑ Dell OptiPlex (source computer 2)
Dell OptiPlex
Dell XPS
Ⓐ
Ⓑ
②
①
②
①
L
L
L
L

Fig. 6. Experiment setup. We take two different kinds of computers — Dell XPS and Dell OptiPlex — as source computers labeled as A and B, respectively. The leftmost computer is another Dell XPS with the same component and configuration as source computer A and acts as an interfering computer. Similarly, the upper right one is the same as source computer B and acts as an interfering computer.

If it is less than 2Hz, it will be encoded to '00'. Otherwise, it is converted to '01'.

Error correction coding We consider widely-used ECC schemes, including two linear correcting codes — Hamming Code and Golay Code — and one non-linear correcting code — Reed-Solomon Code (RS). Hamming code can encode every 4 binary bits to 7-bit codeword and correct 1 bit error. Golay code scheme, converting 12-bit string to 23-bit codeword, can fix up to 3 error bits. $RS(n,k)$ can correct up to $\lfloor \frac{n-k}{2} \rfloor$ mismatching bits. In our evaluation, we use three kinds of RS schemes — **RS(7,3)**, **RS(15,5)**, and **RS(15,3)**.

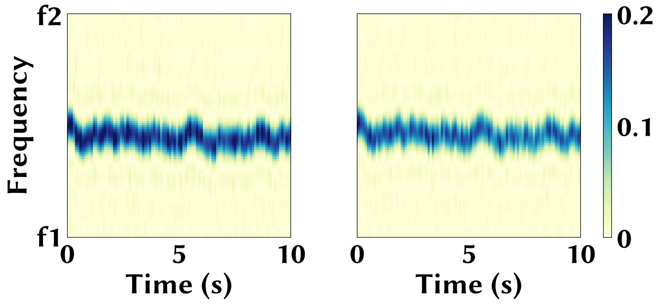
B. Performance Metrics

Entropy is a measurement of randomness of a random variable. Entropy can reflect randomness of keys from the perspective of uncertainty. It is a good indicator for a signal to be a random key generation source. Given a random variable X with n possible values, $X = [x_0, x_1, ..., x_n]$, its entropy can be obtained by $H(X) = -\sum_{i=0}^{V} Pr[x_i] \log_2 Pr[x_i]$, where $Pr[x_i]$ is the probability of the i-th possibility. By encoding adjacent frequency difference, there are three different frequency variations — up, down and still.

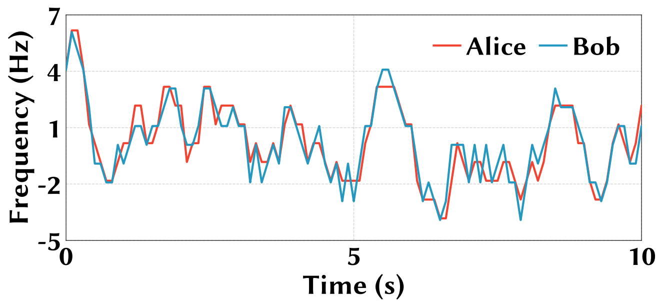
Bit Error Rate (BER) is used to reflect the mismatching level between bits in the same position of two strings. It can be calculated easily by dividing the number of mismatching bits by the total number of bits in the bit string. There are three factors affecting BER in CompKey— FFT time window size, encoding threshold and device distance from the source computer. We will show the impact of these factors using empirical experiments in the next section.

Key Matching Rate (KMR) is also a key metric for secret key generation. It can be calculated by dividing the number of matching keys by the total number of keys. In our experiment, we consider a pair of keys each composed of 60 bits as matching keys if there is no bit discrepancy in any bit position between the two keys.

Bit Generation Rate (BGR) is the number of valid bits generated per second. The higher BGR, the quicker the authentication process finishes and the better the user's experience is. In CompKey, there are three factors determining the BGR. The first one is the FFT time window size Δt, which determines how long it takes to extract the frequency information of each step. The second one is the number of varying possibilities of

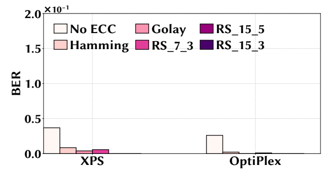


(a)

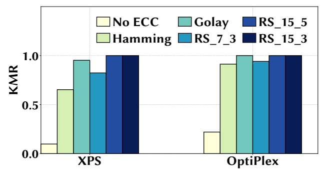


(b)

Fig. 7. (a) Spectrogram of Alice's and Bob's EMR spike in frequency window $[f_1, f_2]$ over 10 seconds. f_2-f_1 is 100Hz. (b) Comparison of frequency variation pattern between Alice and Bob.



(a) Bit Error Rate



(b) Key Matching Rate

Fig. 8. BER and KMR of two experiments with Dell XPS and Dell OptiPlex as source computers, respectively.

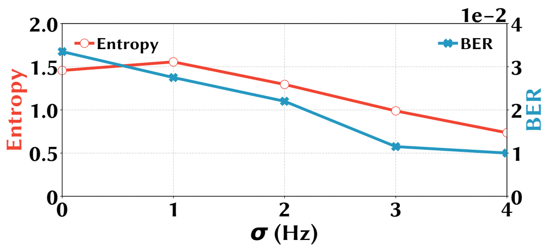


Fig. 9. Different encoding thresholds result in different entropies and BERs.

the EMR signal. More possibilities means more bits generated at a time. Since we decide to encode frequency difference, there are three variations, which need at most two bits to represent. The third one is the choice of ECC. In order to get rid of the information leakage, we need to shrink the size of the bit string by a factor k/n. Based on all these, an equation is given to compute BGR of CompKey: $BGR = \frac{n\Delta t}{k} \log_2 V$, where $V = 3$ in our case.

VI. Evaluation Results

A. Performance of CompKey

We set up two experimental scenarios by using source computers labelled as **A** and **B** in Fig. 6, respectively. We will first introduce the first experiment where we use the desktop Dell XPS (computer A in Fig. 6) as EMR source. This desktop is with another same Dell XPS 1.5m away on its left hand side and with a Dell OptiPlex 9020 1.5m away on the right hand side, as shown in Fig. 6. The interfering Dell XPS is normally used by its owner, who is surfing the Internet. We place Alice 0.5m in front of the source XPS and Bob 0.5m away on the left side of the source computer. Thus, Bob is closer to the interfering XPS and will suffer from more interference. Alice and Bob synchronously collect the EMR signals. In our encoding step, we use 2Hz encoding threshold and take 0.08s as time window size.

By using CompKey, both Alice and Bob extract the frequency of the most prominent spikes from the received EMR signals generated by the source computer. The interfering computer generates weaker EMR than the source computer, and hence its EMR spikes will not be picked up by CompKey. We present the spectrogram of Alice's and Bob's EMR signals during 10 seconds, which is shown in Fig. 7. To compare their frequency change pattern, we put the normalized frequency together (Fig. 7(b)). The frequency changing patterns of Alice and Bob, obtained by CompKey, are highly correlated despite the presence of interfering computers nearby.

As shown in Fig. 6, we set up a second testing scenario and take desktop Dell OptiPlex in the middle as the source computer. Alice and Bob are put within 0.5m away from source computer and Bob is closer to the interfering computer.

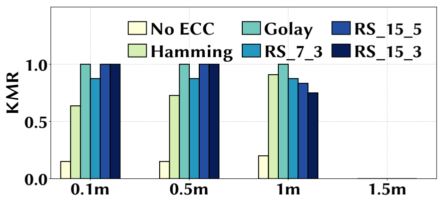
Fig. 8 gives the results of our two experiments, showing that under both testing scenarios CompKey achieves a 100% KMR and demonstrating the practical feasibility of CompKey in the presence of interfering EMR signals.

TABLE II
Randomness Test

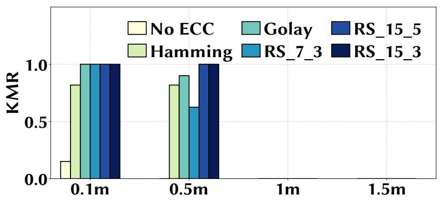
Test	P-value
Frequency	0.936212
Freq. within Block	0.997614
Binary Matrix Rank	0.176145
Non-overlapping Matching	0.780064
Overlapping Matching	0.633007
Linear Complexity	0.029633
Cumulative Sums (Forward)	0.993956
Cumulative Sums (Reverse)	0.993426

B. Randomness of Secret Key

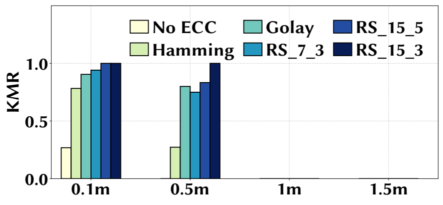
We execute the statistical test suite provided by National Institute of Standards and Technology (NIST) to evaluate the randomness of our generated secret keys [28]. Specifically, if the P-value is more than 1%, the sequence is considered having a high quality of randomness and passing this random-



(a) 100%

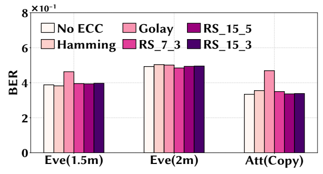


(b) 60%

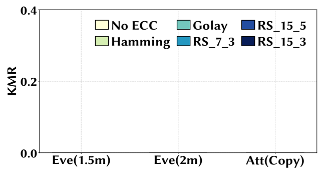


(c) 20%

Fig. 10. KMR with different distances when RAM access frequency of the source computer varies from 100% to 20%.



(a) Bit error rate



(b) Key matching rate

Fig. 11. Performance of attackers.

ness test. Our generated key can pass the statistical tests. The results obtained in our experiment are shown in Table II.

C. Sensitivity of CompKey

1) Impact of Parameters: We see how the encoding threshold σ and time window size Δt affect the performance.

- **Encoding Threshold σ.** When σ is too small, say zero, CompKey will be very sensitive to local noise, which will easily cause mismatching bits. However, if σ is too large, CompKey is more resistant to environment noise but will convert most of the situations to unchanged/still, which reduces the original entropy. We show in Fig. 9 the entropy and BER with respect to different encoding thresholds. Entropy rises a little bit and then decreases. That little increasing entropy at 1Hz may be because when the encoding threshold is 1Hz, some cases with minor changes are converted to unchanged/still, which makes the distribution of three variations more even and hence increases the entropy value. In order to get a small BER and a comparable large entropy, we decide the σ to be 2Hz as a default value.
- **Time Window Size.** With a smaller Δt, less energy will be collected for each EMR signal segment, which will result in more erroneously estimated spikes and lead to a high BER. Nonetheless, a larger Δt reduces the BGR. We calculate the BER, KMR and BGR with respect to different FFT time window sizes. Our results show that 0.08s is a good FFT window size to maintain a high BGR and a low BER. They are omitted due to space limitation.

2) RAM Access Frequency of Source Computer: The more frequently RAM is accessed, the more EMR energy. Typically, a computer's RAM is accessed for 20–60% of the time, depending on how many active and background programs are running. To control the RAM access frequency for experiment, we create an array and load it into the RAM. By controlling how frequently we create arrays, we can manually control the RAM access frequency. For each access frequency of the source computer, we place participating devices at different distances from the source computer and present the KMR.

Fig. 10 shows the KMR with different RAM access frequencies. More RAM activities in the source computer will make more current flowing through memory bus. Therefore, for 100% RAM access frequency, CompKey can reach 100% KMR when devices are 1m away from the source computer. When the RAM access frequency is 20% or 60%, the two users can extract the same secret key with 100% probability when they are both within 0.5m away from the source computer. We also test the most extreme situation when the RAM is forced to be completely idle with no activities, and no secret keys are successfully generated. In practice, however, computers are rarely completely idle as they run multiple background programs, yielding some RAM activities and hence EMR signals. Thus, CompKey can successfully generate secret keys as long as the legitimate devices are put 0.5m within the source computer, whereas the existing WiFi-based approaches require a distance of a few centimeters between legitimate devices [9].

3) Direction: We also evaluate the impact of the direction/angle between legitimate devices and the source computer on CompKey. Our results show that it has little impact on the BER and KMR, and hence are omitted for space limitation.

D. Security Analysis

For eavesdropping, we consider two Eves, who are capturing the EMR 1.5m and 2m away from the source computer, respectively. We use Dell XPS as the source computer and set its RAM access frequency to be 60% (a fairly strong one to favor attackers). Let the legitimate users and attackers collect the EMR signals at the same time. When Eve is 1.5m away, he can still get some frequency information about the EMR. However, the Eve who is 2m away can get nothing, thus making r_2 =2m a safe threshold distance for attackers in our threat model (shown in Fig. 1). The resulting power spectrum density (PSD) of legitimate users and two Eves are omitted due to space limitation.

For copy attacks, we use one Dell XPS computer to play a video and two legitimate devices collect EMR from this computer 0.5m away. Meanwhile, we use another identical computer to play the same video and collect the EMR emitted from it at the same time. After collecting the EMR signals, we follow CompKey to generate the secret key and get the performance with different ECCs.

Fig. 11 gives the performance of two kinds of attackers. We can see that even if attackers know all the detailed information, they still cannot get the accurate secret key and the KMR is practically zero (even when Eve is 1.5m away from the computer). This demonstrates the security of CompKey against both eavesdropping and copy attackers.

VII. Related Works

There are numerous studies on proximity-based authentication by extracting shared secret keys from ambient signals. Here, we review some of the most related ones.

In radio-based authentication studies, received signal strength (RSS) [6], [11], [29] and channel state information (CSI) [12], [30], [4], [13], [5], [12] are two widely-used random signal attributes. However, since only one RSS value can be extracted from one WiFi packet, RSS-based key generation methods are subject to low BGR. Other studies adopt CSI of radio channel for authentication [5], [12], [4]. However, both RSS-based and CSI-based methods only work for a limited authentication distance because two devices must be placed close to each other to sense the same signal amplitude attribute.

Audio-based authentication approaches make use of the characteristics of acoustic channel [14], [15], [16]. Nonetheless, it takes time to get the statistics of acoustical attributes, which makes the method subject to low BGR. While key generation based on acoustic channel response (ACR) can help improve BGR, it applies to only two parties to exchange a probe sound. Biometrics is another widely used authentication approach, which utilizes signals from human body for secret key generation [7], [31], [32]. Since this kind of method needs to capture special signals like body potential signal, it requires the participating devices be equipped with special sensors.

VIII. Conclusion

In this paper, we propose CompKey to secure wireless D2D communications. We observe that the memory bus inside a computer can emit EMR and that only devices in the vicinity of the computer can reliably extract frequency information from the signal. CompKey employs a novel difference-based scheme to encode the frequency variation of computer EMR to a bit string and adopts reconciliation method to alleviate the discrepancy between two bit strings. Through evaluation, we show that devices within 0.5m away from the computer can get identical keys with 10 bits/s BGR and 100% KMR.

Acknowledgement

F. Yang and S. Ren were supported in part by the U.S. NSF under grant CNS-1551661.

References

[1] L. Song, D. Niyato, Z. Han, and E. Hossain, *Wireless Device-to-Device Communications and Networks*. New York, NY, USA: Cambridge University Press, 2015.

[2] Asadi, Arash, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," in *IEEE Communications Surveys & Tutorials 16*, 2014.

[3] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "On the effectiveness of secret key extraction from wireless signal strength in real environments," in *MobiCom*, 2009.

[4] X. Wei, C. Qian, J. Han, K. Zhao, S. Zhong, X.-Y. Li, and J. Zhao, "Instant and robust authentication and key agreement among mobile devices," in *CCS*, 2016.

[5] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam, "Proximate: proximity-based secure pairing using ambient wireless signals," in *MobiSys*, 2011.

[6] A. Varshavsky, A. Scannell, A. LaMarca, and E. de Lara, "Amigo: Proximity-based authentication of mobile devices," in *UbiComp*, 2007.

[7] L. Yang, W. Wang, and Q. Zhang, "Secret from muscle: Enabling secure pairing with electromyography," in *SenSys*, 2016.

[8] W. Stallings, "Cryptography and network security: principles and practice," in *Upper Saddle River: Pearson*, 2017.

[9] J. Zhang, T. Q. Duong, A. Marshall, and R. Woods, "Key generation from wireless channels: A review," *IEEE Access*, vol. 4, pp. 614–626, 2016.

[10] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theor.*, vol. 22, pp. 644–654, Sept. 2006.

[11] H. Shafagh and A. Hithnawi., "Come closer: proximity-based authentication for the internet of things," in *MobiCom*, 2014.

[12] H. Liu, Y. Wang, J. Yang, and Y. Chen, "Fast and practical secret key extraction by exploiting channel response," in *Proceedings IEEE INFOCOM*, pp. 3048–3056, IEEE, 2013.

[13] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel," in *MobiCom*, 2008.

[14] D. Schürmann and S. Sigg, "Secure communication based on ambient audio," in *IEEE Transactions on mobile computing 12.2*, pp. 358–370, 2011.

[15] P. Xie, J. Feng, Z. Cao, and J. Wang, "Genewave: Fast authentication and key agreement on commodity mobile devices," *IEEE/ACM Trans. Netw.*, vol. 26, pp. 1688–1700, Aug. 2018.

[16] N. Karapanos, C. Marforio, C. Soriente, and S. Capkun, "Sound-proof: Usable two-factor authentication based on ambient sound," in *USENIX Security*, 2015.

[17] M. Miettinen, N. Asokan, T. D. Nguyen, A.-R. Sadeghi, and M. Sobhani, "Context-based zero-interaction pairing and key evolution for advanced personal devices," in *CCS*, 2014.

[18] Z. Luo, W. Wang, J. Xiao, Q. Huang, T. jiang, and Q. Zhang, "Authenticating on-body backscatter by exploiting propagation signatures," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, pp. 123:1–123:22, Sept. 2018.

[19] W. Wang, L. Yang, Q. Zhang, and T. Jiang, "Securing on-body iot devices by exploiting creeping wave propagation," *IEEE Journal on Selected Areas in Communications*, vol. 36, pp. 696–703, April 2018.

[20] M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "Revisiting context-based authentication in iot," in *DAC*, 2018.

[21] J. Wan, A. B. Lopez, and M. A. Al Faruque, "Exploiting wireless channel randomness to generate keys for automotive cyber-physical system security," in *ICCPS*, 2016.

[22] Intel, "Intel mobile communications software-defined radio," https://www.intel.com/content/www/us/en/products/docs/wireless-products/mobile-communications/software-defined-radio.html.

[23] S. Lu, J. Yuan, S. Yu, and M. Li, "ask-ban,"

[24] M. Guri, A. Kachlon, O. Hasson, G. Kedma, Y. Mirsky, and Y. Elovici, "Gsmem: Data exfiltration from air-gapped computers over GSM frequencies," in *USENIX Security*, 2015.

[25] Wikipedia, "Hardware random number generator," https://en.wikipedia.org/wiki/Hardware_random_number_generator#Clock_drift.

[26] Low Electromagnetic Emissions Office Equipment, "Electromagnetic emissions of desktop computers," https://www.lowemfoffice.com/desktop_computers.htm.

[27] K. S. Qiao, Yue and A. Arora, "Shape matters, not the size: A new approach to extract secrets from channel," in *HotWireless*, 2014.

[28] L. Bassham, "A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications," in *NIST SP*, pp. 800–22rev1a, 2010.

[29] A. Kalamandeen, A. Scannell, E. d. Lara, A. Sheth, and A. LaMarca, "Ensemble: cooperative proximity-based authentication," in *MobiSys*, 2010.

[30] X. Wei, X.-Y. Li, C. Qian, J. Han, S. Tang, J. Zhao, and K. Zhao, "Keep: Fast secret key extraction protocol for D2D communication," in *IWQoS*.

[31] J. Lester, B. Hannaford, and G. Borriello, "Are you with me? – using accelerometers to determine if two devices are carried by the same person," in *International Conference on Pervasive Computing*, 2004.

[32] W. Wang, L. Yang, and Q. Zhang, "Touch-and-guard: secure pairing through hand resonance," in *UbiComp*, 2016.