Instant and Robust Authentication and Key Agreement among Mobile Devices

Wei Xi[1], Chen Qian[2], Jinsong Han[1], Kun Zhao[1], Sheng Zhong[3], Xiang-Yang Li[4], Jizhong Zhao[1]
[1]Xi'an Jiaotong University; [2]University of California Santa Cruz;
[3]Nanjing University; [4]University of Science and Technology of China;
{weixi.cs, hanjinsong, xiangyang.li}@gmail.com;cqian12@ucsc.edu;
zhongsheng@nju.edu.cn; {zhaokun2012, zjz}@mail.xjtu.edu.cn

ABSTRACT

Device-to-device communication is important to emerging mobile applications such as Internet of Things and mobile social networks. Authentication and key agreement among multiple legitimate devices is the important first step to build a secure communication channel. Existing solutions put the devices into physical proximity and use the common radio environment as a proof of identities and the common secret to agree on a same key. However they experience very slow secret bit generation rate and high errors, requiring several minutes to build a 256-bit key. In this work, we design and implement an authentication and key agreement protocol for mobile devices, called The Dancing Signals (TDS), being extremely fast and error-free. TDS uses channel state information (CSI) as the common secret among legitimate devices. It guarantees that only devices in a close physical proximity can agree on a key and any device outside a certain distance gets nothing about the key. Compared with existing solutions, TDS is very fast and robust, supports group key agreement, and can effectively defend against predictable channel attacks. We implement TDS using commodity off-the-shelf 802.11n devices and evaluate its performance via extensive experiments. Results show that TDS only takes a couple of seconds to make devices agree on a 256-bit secret key with high entropy.

Keywords

Group authentication; Key agreement; WiFi; CSI

1. INTRODUCTION

With the rapid technology growth of mobile devices, wireless device-to-device communication has been playing important roles for many emerging applications including Internet of Things (IoT) and mobile social networks. For example, IoT appliances may communicate with each other to collaboratively sense the physical world and make proper reactions [19]. Mobile social applications fuel the need for mobile devices such as smartphones to interact directly in an ad-hoc mode to share various information such as texts, pictures, and videos.

A fundamental problem of wireless device-to-device communication is the vulnerability to various attacks such as identity spoofing, eavesdropping, and man-in-the-middle attacks [24] [15] [4]. Sensitive information such as health conditions and personal data shared among IoT and Mobile social devices has become the targets of these attacks. Hence authentication and key agreement among mobile devices is the critical first step to secure such interactions. It requires a number of mobile devices to agree on a symmetric key without prior shared secret, through an untrusted and unauthenticated wireless channel. The key then helps to establish a secure channel for these devices. Hence *authentication and key agreement are both required* in building the secure channel. This process is also called as device pairing or grouping in the literature.

Traditional public key encryption and Diffie-Hellman key exchange [6] do not work for device-to-device communication in mobile networks due to the open nature of the wireless medium and lack of centralized trust management [16] [8]. Recent efforts have been made to device authentication and key agreement while reducing the amount of user interactions such as manual key assignment and input [24] [15] [23] [19]. The main idea of these methods is to put two devices into physical proximity and use the common radio environment as a proof of identities and the common secret to generate a same key on different devices. The main limitations of these methods is slow speed of key generation and high error rate. For example, Radiotelepathy [16] extracts secret keys using the channel impulse response (CIR) in the wireless channel and its key generation rate is only around 1 bit per second. ProxiMate can generate less than five bits per second in most scenarios [15], costing more than one minute for two devices to agree on a 256-bit key. Holding two devices in a physical proximity (5cm in the ProxiMate

CCS'16, October 24-28, 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2976749.2978298

experiments [15]) for minutes before communication is inconvenience in most applications. A recent work can pair two IoT devices co-present in a long period of time [19]. This solution is good for wearable devices that are attached to the same object or human being, but impractical for devices that may move away from each other. Pinpoint [27] can pair two devices and estimate secure channel leveraging semi-symmetrical CIR to dispel reversed jamming noise. However, this method can hardly expand to more than two devices yet. Moreover, many existing key agreement methods assumes that two devices communicate through an authenticated channel [2] [16] [15] [14] [28]. Without extra device authentication solutions, these methods are vulnerable to various attacks such as a man-in-the-middle attack.

In this work, we design and implement an authentication and key agreement protocol for mobile devices with instant and robust key agreement. Our experiments using commodity off-the-shelf (COTS) wireless devices show secret bit generation rates *faster than existing methods by more than an order of magnitude* in various practical scenarios. The key idea behind the speed improvement is a new key distribution protocol. Different from many existing methods that use received signal strength (RSS) to extract secret bits [15], we use channel state information (CSI) available from Orthogonal Frequency Division Multiplexing (OFDM) of the current WiFi standard. Different from RSS, CSI measurement contains much richer information in a same period of time. On the other hand, previous methods that use *reciprocal quantization* (directly converting each signal sample to a bit) may introduce many mismatched bits for the two keys generated on different devices [2] [16] [15] [14] [28], requiring additional *information reconciliation* process [3] to fix the errors. Moreover, since CSI is very sensitive to location especially in indoor environment, the authentication distance for all existing proximity-based methods (*e.g.*, Amigo [24], ProxiMate [15]) should be less than 0.1λ (1.25cm@2.4GHz) using CSI, which is not practical for WiFi devices. To overcome these limitations, our solution uses substitution-based key delivery instead of quantization-based key extraction, which is highly robust for secret bit agreement. We name this method as The Dancing Signals (TDS).[1]

Besides fast key generation, another unique feature of TDS is that the secret key can be an arbitrary bit string specified by one of the devices, while in existing methods the agreed key completely depends on common wireless channel information. This feature brings three important advantages: i) TDS can always build a *key with strong randomness* and avoid keys with low entropy [8]. ii) TDS can built a key among *more than two devices*. In previous pairing methods, it is hard for more than two devices to simultaneously generate a same key, since mismatched bits between every pair of keys lead an unaffordable agreement overhead and significant risk of key leakage. TDS allows a key to be directly delivered from one device to others, saving huge amount of overhead from interactive agreement. iii) TDS is very *robust to the predictable channel attack* [8]. In such an attack, an adversary uses planned movements to cause desired and predictable changes in the channel between the two devices and further predict the key generated from the channel.

We summarize the advantages of TDS as follows:

- TDS achieves both device authentication and key agreement. Compared with prior methods that only focus on one of them such as [1] [23] [28], TDS is more robust to various attacks.
- The secret bit generation rate of TDS is faster than existing solutions [24] [15] by over an order of magnitude. Our implementation on COTS devices show generation rates of hundreds bits/sec.
- Previous device pairing protocols can only support two devices. TDS works well for more devices.
- TDS can be used to transmit any confidential bit sequence specified by the sender, including self-generated session keys, which avoids keys with low entropy.
- TDS can effectively defend against predictable channel attacks.

The rest of this paper is organized as follows. Section 2 presents the system model and observations from our experiments. Section 3 details the protocol design of TDS. We provide the analysis for the security and efficiency of TDS in Section 4. Section 5 shows the evaluation results based on the implementation of TDS on off-the-shelf mobile devices. We present the related work in Section 6 and conclude our work in Section 7.

2. SYSTEM MODEL AND OBSERVATION

In this section, we first define the system and security model of TDS. We then use analysis and experiments to demonstrate the feasibility and challenges of using CSI measurement to make key agreement among multiple devices.

2.1 System and security model

We assume that multiple legitimate wireless devices, Alice, Bob, and Calvin, are interested in securely exchanging their private information. They are able to communicate via the standard IEEE 802.11 protocols with OFDM, such as WiFi. They have no prior shared secret. When performing key agreement, the devices need to be placed by their users in a physical proximity such that the distance from any device to Alice is less than a *authentication distance* ($0.4\lambda \approx 5cm$@2.4GHz where λ is the wavelength). A malicious device Eve is located beyond a *safe distance* ($\lambda \approx 12.5cm$ for WiFi) to Alice. If Eve moves into the safe distance, it will be easily seen by the users of Alice and Bob. Eve can sense the wireless environment, inject new traffic, and replay packets. Alice, Bob, Calvin and Eve can hear a public wireless source Peter. Eve can perform various attacks including spoofing, eavesdropping, and man-in-the-middle. In the most extreme case, Eve may control Peter that Alice, Bob and Calvin are using for key agreement. Then Eve can turn the signal strength into any pattern she desires. Eve has complete knowledge of the proposed method and algorithms.

The goal of this system is to *instantly* make Alice, Bob, and Calvin agree on a strong symmetric key without letting

[1]The name was inspired by the story "The Adventure of the Dancing Men" written by Arthur Conan Doyle. In the story Sherlock Holmes receives a paper with a sequence of dancing men figures from a client. He later realizes each dancing man is a substitution of an English letter and cracks the code.

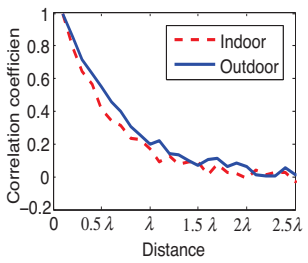


Figure 1: Pearson correlation coefficient decreases with growing distance

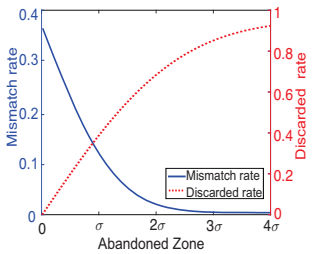


Figure 2: Dilemma of mismatch rate and discarded rate

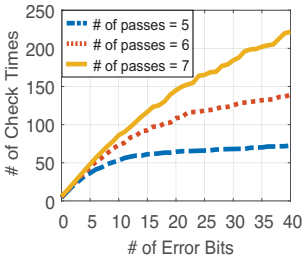


Figure 3: Mismatched bits cause high cost in error correction

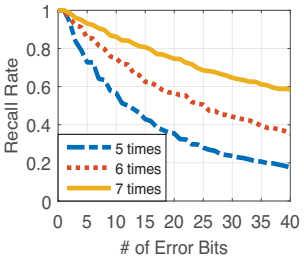


Figure 4: Information reconciliation fails more seriously with increasing error bits

Eve know about the key. We have the following security requirements.

- *Authenticity.* A device needs to ensure that it is making key agreement with other legitimate devices. Any spoofer will be detected.
- *Confidentiality.* Any information of the key should not be exposed to Eve.
- *Integrity.* The key should be consistent at all legitimate devices.

We *do not consider availability* in this work. If Eve block all WiFi signals, devices may not agree on any key.

2.2 Feasibility of CSI-based key generation

In this paper, we use CSI as the proof of authenticity and source of common secret information. The intuition of using CSI is that it is a unique and correlated measurement for devices around a particular physical location. CSI measurements at different devices are rapidly de-correlated with distance between them. In addition, CSI is unpredictable due to its random property caused by the multipath effect of signal propagation. CSI is a much richer source of secret information than the one of RSS, because it contains the information of 56 subcarriers in each measurement sample.

We demonstrate the properties using experimental validation. We use two laptop computers Alice and Bob, each equipped with COTS wireless NIC model Intel 5300 operating in the 802.11n 2.4GHz channel.[2] They then collect measurement results of the CSI amplitude values independently from a public WiFi while varying their distance from 0.1λ to 2.5λ, where the wavelength $\lambda = 12.5$cm for 2.4GHz. Figure 1 shows the Pearson product-moment correlation coefficient of the CSI samples from the two devices. We found that when the distance is smaller than 0.5λ, the samples are highly correlated. The correlation drops quickly with the distance growth. When the distance $> 2\lambda$, the samples are uncorrelated.

The above properties of CSI are important for device authentication and key agreement. Only if the samples from different devices nearby are similar, CSI can be a proof of physical proximity. Only if the samples are rapidly de-correlated with distance between the devices, CSI can be a common secure information.

[2]We use laptops for the ease of programming. The method can be applied to any devices with 802.11 NICs.

2.3 Challenges of CSI-based key generation

Suppose two devices, Alice and Bob both listen to a public WiFi source. For each of them, the CSI amplitude value $h(t)$ at time t can be directly obtained from an existing API of the Intel 5300 network card. To extract secret information from two similar measurements of CSI amplitude values, a simple approach is to determine a cut-off amplitude level h and use 0 to represent samples smaller than h and 1 to represent samples larger than h. For example, h can be 0.5 for CSI amplitude varying in $[0, 1]$. This method is called *reciprocal quantization*.

Reciprocal quantization may cause mismatched bits at two devices. For example, if Alice gets a CSI value 0.53 for a particular bit, Bob gets 0.48, and the cut-off is 0.5, then they will have a different bit. To reduce these mismatched bits, existing quantization methods often use an abandoned zone. For example, if the abandoned zone is $[0.4, 0.6]$, then only if a CSI value is less than 0.4 (or larger than 0.6), it can be converted to a 0-bit (or 1-bit).

Selecting the size of the abandoned zone is a dilemma: if the zone is small, mismatched bits still occur; if the zone is large, too many CSI samples will be discarded, slowing secret bit generation. Figure 2 shows the bit mismatch rate versus the discarded bit rate by varying the abandoned zone from 0 to 4σ, where σ is the standard deviation of the Gaussian noise. We find that when the abandoned zone is smaller than σ, the discarded rate is low but it causes more than 10% mismatched bits. When the zone is large, *e.g.*, 3σ, the mismatch rate is negligible but more than 80% samples will be discarded. To further demonstrate the harm of mismatched bits, we use an existing method, information reconciliation [3] [8], to fix mismatched bits by iterative parity checks. Figure 3 shows that the rounds of parity checks increase significantly with growing mismatched bits, for generation of a 256-bit key. For 20 mismatched bits, it requires more than 70-150 parity check bits to correct them. Besides tremendous communication and time cost, the number of secret bits is also reduced from 256 to < 150 due to privacy amplification [17]. Additionally, information reconciliation is a

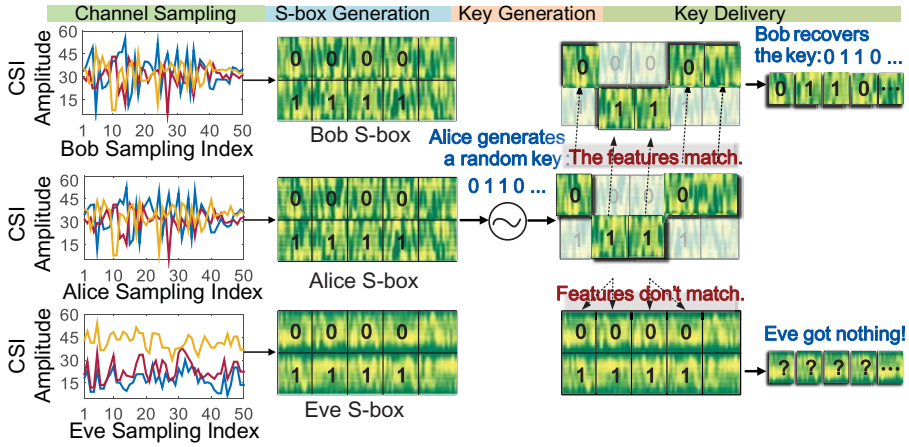


Figure 5: Main steps of TDS: channel sampling, S-box generation, key generation, key delivery. The final step information reconciliation is not shown.

probabilistic technique, it fails occasionally. Figure 4 shows that recall rate of information reconciliation reduces with growing mismatched bits, even a large number of passes is chosen. Therefore 10% mismatched bits for a 256-bit key would cause huge time/communication cost and secret bit loss.

More importantly, in reciprocal quantization, the two devices have no ability to decide which bits to generate. In some cases, the generated key may have low entropy [8]. In addition, it has been observed that near-by subcarriers of OFDM may have correlated CSI measurements [28], which reduces the security level of the generated keys.

As a conclusion, directly converting CSI or RSS sample values to secret bits does not work. It is the reason that some existing work, such as ProxiMate [15] has to use FM and TV signals which have very long wavelengths, rather than WiFi signals, to generate robust secret keys.

3. PROTOCOL DESIGN

We present the design of our protocol TDS in this section.

3.1 Basic idea of TDS

Instead of asking all devices to perform quantization separately, we allow one device, say Alice, to decide an arbitrary key and distribute it to other legitimate devices with confidentiality. Since extracted key of Calvin are identical to Bob's, we use Alice, Bob and Eve to sketch our idea shown in Figure 5, which includes the following steps:

1) Channel sampling. The users first place Alice and Bob and the distance between their NIC cards is $< 5cm$. A user then starts the TDS program on Alice and makes Alice an *initiator*. Alice sends a message to other devices and ask them (including Alice) to start listening to a same public WiFi source. Current WiFi standard uses orthogonal frequency-division multiplexing (OFDM) and there are 56 orthogonal subcarrier signals to carry data on parallel data streams. Hence at a same time, the CSI measurement of a WiFi source includes up to 56 sample values from different sub-carriers. Figure 5 shows samples of amplitude values from three sub-carriers of a same WiFi source. Note that Alice and Bob have highly correlated sample values, but Eve's measurement is very different to theirs.

2) S-box Generation. After obtaining enough number of samples, Alice will ask other devices to synchronize the sampled data. Then each device will construct an S-box, which includes a number of *blocks*. Each block contains a number of samples and represents a bit 0 or 1. Note blocks are organized in pairs. In Figure 5 we show the first four pairs of blocks of each S-box, representing four 0-bits and four 1-bits. Later we will introduce the mechanism that guarantees every legitimate device will generate an S-box in which the samples in the blocks are consistent to Alice's S-box.

3) Key generation. Alice may use any sophisticated key generation method to determine a strong secret key with high randomness and entropy. In the Figure 5 example, she uses a key starting with 0110.

4) Key delivery. For every bit of the key, Alice select a block from every pair to represent whether this bit is 0 or 1. For example, in Figure 5 the first four bits are 0110. Hence Alice selects the first 0-block, the second 1-block, the third 1-block and the fourth 0-block. Then Alice broadcasts these blocks to other devices. Since Bob's measurement is similar to Alice's, Bob can obtain a similar S-box. When Bob receives the blocks sent from Alice, it can easily recover these blocks to a bit stream. Bob only needs to decide whether the i_{th} block is more similar to his i_{th} 0-block or his i_{th} 1-block. Eve, which is out of the safe distance from Alice, cannot obtain an S-box with any correlation to Alice's. Even if Eve can hear all blocks sent from Alice, she is not able to match any block to a 0-bit or 1-bit.

5) Information reconciliation. Finally, Alice and Bob need to ensure that they obtain a same key and correct the mismatch bits, which are very few in TDS. TDS uses an information reconciliation method, as presented in prior work [3, 28]. Note that the protocol has a threshold T such only

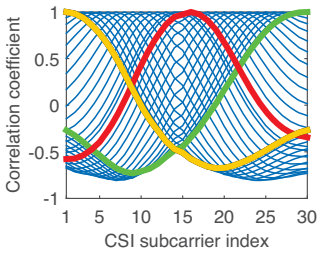


Figure 6: CSI correlations

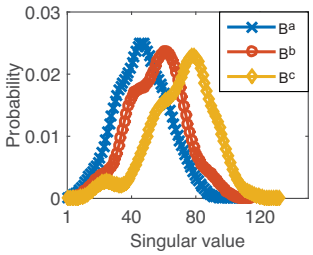


Figure 7: Distributions of $\hat{\sigma}_2$

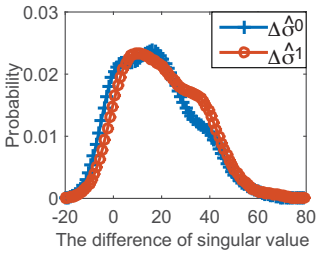


Figure 8: Distributions of $\Delta\hat{\sigma}_2$

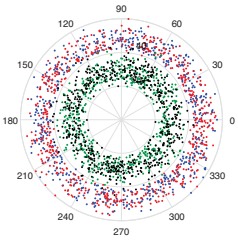


Figure 9: Feature distributions of two group sets

a device with error bits fewer than T can start information reconciliation with Alice.

3.2 Sampling and S-box generation

All devices measure the CSI samples simultaneously using an existing synchronization protocol. Note that complete synchronization is not necessary, as long as all three devices can have a large number of CSI samples in common. For example, Alice can send a sequence of samples and ask all devices to construct their S-boxes after this sequence.

3.2.1 Block and group allocation

Block allocation divides all CSI measurement samples into blocks representing 0 and 1 bits. Therefore if Alice wants to deliver a 256-bit key, she should construct at least 256 pairs of blocks first and then use a block from every pair to present the bit value. The intuition to use a block of samples rather than a single one is to reduce the mismatch rate. The block size n influences the performance of key delivery. A small n leads to unstable blocks whose features are prone to the ambient noise, while a large n reduces the efficiency of key delivery. Based on our empirical results, we select $n = 6$ in our implementation for WiFi.

For OFDM signals, each sample includes m subcarriers and each subcarrier has one CSI value. Therefore, one block has $m * n$ CSI values, which are divided into two groups. And then, the features of these two groups represent 0 and 1 respectively. Group allocation is challenging due to the following two main requirements.

Reliability Requirement (R1): The features of two groups in a same block, representing 0 and 1 respectively, should be sufficiently distinct to each other, to avoid mismatched bits.

Security Requirement (R2): The features from all 0-groups and 1-groups should be identically distributed across different groups. Otherwise given a feature, an eavesdropping could improve its guess on the bit by studying the distributions of 0-groups and 1-groups.

We reuse each measurement sample in S-box generation to improve the utilization of measurement results and reduce channel measurement time. Assuming there are N samples, we use the following construction method:

1. Every n successive samples are put into one block, for $i \in \{1 : \lfloor \frac{N}{n} \rfloor\}$.

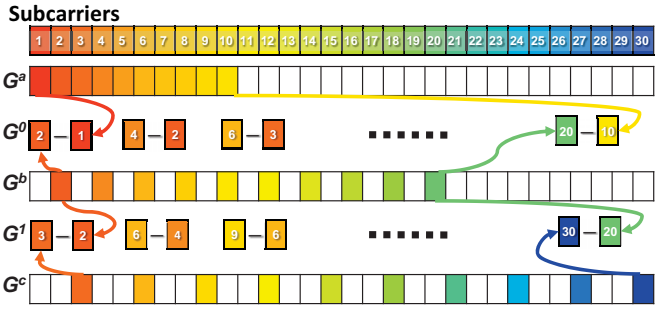


Figure 10: Group allocation

2. In addition, for $j = 1 : \lfloor \frac{N}{n} \rfloor$, the j_{th}, $(n+j)_{th}$, $\cdots$ and $((\lfloor \frac{N}{n} \rfloor - 1)n + j)_{th}$ samples are put into one block.

The number of blocks is about $2N/n$. Using Intel 5300 NIC, each sample includes 30 CSI amplitude values from 30 subcarriers.[3] Hence each block is a $30 \times n$ matrix.

To represent 0 and 1, we divide a block into two groups. In each block, we denote the measurements of the i_{th} subcarrier as S_i, a vector including n values. An intuitive solution is to let $G^0 = \{S_1, S_3, S_5, .., S_{29}\}$ and $G^1 = \{S_2, S_4, S_6, .., S_{30}\}$, where G^0 represents 0 and G^1 represents 1. We find that this allocation does not satisfy the reliability requirement R1. From our experiments, we find an important property of CSI samples from different subcarriers. The adjacent subcarriers have very strong CSI correlation, and the correlation oscillates with increasing the difference of two subcarriers' indexes. Figure 6 shows the correlation between two different subcarriers. The yellow curve plots the CSI correlation between the 1st subcarrier and the x_{th} subcarrier. The 1st subcarrier has strongest correlation with 2nd subcarrier, and has little correlation with 11_{th} subcarrier. Red and green curves plot the correlation about 15_{th} and 30_{th} subcarriers to the x_{th}h subcarrier respectively. The difference between any pair of subcarriers that are with a fixed difference of indexes is very close. For example, subcarrier pairs $\{2, 4\}$ and $\{7, 9\}$ will have a similar difference, *i.e.*, $S_2 - S_4 \approx S_7 - S_9$.

Given the above observations, the group allocation $G^0 = \{S_1, S_3, S_5, .., S_{29}\}$ and $G^1 = \{S_2, S_4, S_6, .., S_{30}\}$ will result in similar G^0 and G^1. It is because S_1 and S_2 are very

[3]Technically there are 56 subcarriers (52 data subcarriers and 4 pilots) for 802.11n, but the current CSI tool can only provide 30 of them.

close, S_3 and S_4 are close, and so on. As the consequence, the groups for 0-bit and 1-bit are not distinct. Similarly, the group allocation $G^0 = \{S_1, S_2, .., S_{15}\}$ and $G^1 = \{S_{16}, S_{17}, .., S_{30}\}$ is also not acceptable, because $S_{16} - S_1 \approx S_{17} - S_2 \approx .. \approx S_{30} - S_{15}$.

We then attempt to select subcarriers with varying intervals among them. For every block, we construct three groups, $G^a = \{S_1, S_2, S_3, \cdots, S_{10}\}$, $G^b = \{S_2, S_4, \cdots, S_{20}\}$, and $G^c = \{S_3, S_6, S_9, \cdots, S_{30}\}$. These three groups show significant difference and then can be used to represent different bit values.

3.2.2 Feature extraction

Each group G^a, G^b, or G^c is then an $m \times n$ matrix. In our implementation $m = 10, n = 6$. To efficiently deliver the secret key to other devices, Alice will send a feature representing the block of the bit rather than the entire matrix. Due to the noise interference, CSI variations among Alice and other close devices always exist. In TDS, we leverage the singular value decomposition (SVD) to solve this issue. SVD provides a convenient way to characterize a matrix. Each group G is expressed as $G_{m\times n} = U_{m\times m}\hat{\Sigma}_{m\times n}V^T_{n\times n}$, where the diagonal matrix $\hat{\Sigma}$ is uniquely determined by G. The diagonal elements of $\hat{\Sigma}$, $\hat{\sigma}_1, \hat{\sigma}_2, ..., \hat{\sigma}_n$ (assuming $n \leq m$), are called singular values. In TDS, we extract the feature from $\hat{\Sigma}$ to characterize each group. It is well known that large singular values preserve the internal properties of a matrix in a low dimensional space [9]. We propose to use the second and third largest singular values in each group, denoted as $\hat{\sigma}_2$ and $\hat{\sigma}_3$, as the feature of the group that will be broadcast by Alice instead of the whole matrix. We do not use the largest singular values in each group because empirical results show that they are hard to differentiate. After receiving $\hat{\sigma}_2$ and $\hat{\sigma}_3$ from Alice, Bob will compare them with the singular values of his G^0 and G^1 groups and determine whether this bit is 0 or 1. The small singular values is mainly due to noise in the data, which are discarded.

Unfortunately, the above features cannot satisfy the security requirement. Figure 7 plots the distributions of singular value $\hat{\sigma}_2$ of three types of blocks G^a, G^b, and G^c. We find that their distributions are distinct, although there is still a large overlapped area. However, utilizing this knowledge, an eavesdropper can improve its guess on the value of this bit, given the singular value it received. For example if it receives a large singular value $\hat{\sigma}_2$, then it is more likely to represent G^c.

3.2.3 Final feature computation

The final solution to select a feature that satisfying all requirements specified in Subsection 3.2.1 is presented as follows. For every block of samples constructed, we first compute G^a, G^b, and G^c and compute their 2nd and 3rd largest singular values: $\hat{\sigma}^a_2$, $\hat{\sigma}^a_3$, $\hat{\sigma}^b_2$, $\hat{\sigma}^b_3$, $\hat{\sigma}^c_2$, and $\hat{\sigma}^c_3$. We let two difference values $(\Delta\hat{\sigma}^0_2, \Delta\hat{\sigma}^0_3)$ to represent the bit 0 and $(\Delta\hat{\sigma}^1_2, \Delta\hat{\sigma}^1_3)$ to represent the bit 1 as shown in Figure 10, where $\Delta\hat{\sigma}^0_2 = \hat{\sigma}^b_2 - \hat{\sigma}^a_2$, $\Delta\hat{\sigma}^0_3 = \hat{\sigma}^b_3 - \hat{\sigma}^a_3$, $\Delta\hat{\sigma}^1_2 = \hat{\sigma}^c_2 - \hat{\sigma}^b_2$, and $\Delta\hat{\sigma}^1_3 = \hat{\sigma}^c_3 - \hat{\sigma}^b_3$. Figure 8 shows the distributions of $\Delta\hat{\sigma}^0_2$ and $\Delta\hat{\sigma}^1_2$. We can find that their distributions are almost identical. The distributions of $\Delta\hat{\sigma}^0_3$ and $\Delta\hat{\sigma}^1_3$ are also identical, which is not shown. In this way, the attacker cannot improve its guess of a bit based on the feature sent from Alice.

3.3 Key generation and delivery

Key generation. Alice, the initiator, is in charge of generating a key with strong randomness, using any existing algorithm. The bit string for the secret key should be (1) *sufficient long*, *i.e.*, more than 128 bits in common, and (2) *statistically random*.

Feature pairing. After computing the features for 0/1 bits and generating the key, TDS needs to perform feature pairing, *i.e.*, choosing one from features of 0-bit and one from features of 1-bits and making them a pair to represent a bit. The simplest solution is to make the two features computed from a same group of samples to be a pair. However, one disadvantage is that it is possible that in some pairs the two features are close and hence make it easier to produce mismatched bits. Hence this feature pairing step is to find an optimal strategy of making the difference of feature values for each 0/1 pair larger than a certain threshold θ.

We take a paring algorithm based on Max-Weighted Bipartite Matching to solve this problem. We assemble all the features extracted from G^0 and G^1 into two sets C^0_f and C^1_f to represent 0 and 1 bits respectively. The above problem can be formalized as a Max-Weighted Bipartite Matching problem. Then we can leverage Kuhn-Munkras algorithm [10] to solve this problem, and map the 0 and 1 bits to the features.

We construct a complete bipartite graph $G(C^0_f, C^1_f, C^0_f \times C^1_f)$ with weights $w(e_{ij}) = |c^0_{f_i} - c^1_{f_j}|$. The feasible vertex labeling l is defined as

$$\begin{cases} l(c^0_{f_i}) = \max w\left(c^0_{f_i}, c^1_{f_j}\right) & \forall c^0_{f_i} \in C^0_f \\ l(c^1_{f_j}) = 0 & \forall c^1_{f_j} \in C^1_f \end{cases}$$

The Equality Subgraph G_l is a spanning subgraph of G which is defined as

$$G_l = \{(x_i, y_j) | G_l \subseteq G, w(x_i, y_j) = l(x_i) + l(y_i)\} \quad (1)$$

where

$$\begin{cases} x_i \in X \subseteq C^0_f \\ y_j \in Y \subseteq C^1_f \end{cases}$$

The algorithm execution time is much shorter than CSI measurement time.

Key delivery and information reconciliation. To represent the sequence of generated bits K_a whose length is l, Alice selects l features from l pairs in her S-box S_a. She then sends the features to Bob and Calvin. Bob and Calvin use their S-boxes S^{-1}_b and S^{-1}_c to decode the key. TDS uses existing information reconciliation method [3] [28] to create consistent keys on difference devices. After information reconciliation, Bob's key will be corrected to Alice's. If there are more than two devices, Alice will run information reconciliation to every of them in order. Information reconciliation includes an information-theoretically secure authentication using universal hashing [17]. Secure authentication can also defend against impersonation attack, in which an attacker pretends to be Alice and send a sequence of feature values, and substitution attack, in which an attacker replace the correct feature values by other values. The process is

Algorithm 1: KM based feature mapping

Input: C_f^0,C_f^1,θ;
Output: Perfect matching bipartite graph $\mathbf{G_L}$ with maximum difference

Build Equality Subgraph $\mathbf{G_L}$ via Equation (1);
Find the maximum matching graph G_l utilizing Hungarian algorithm [10];
for $\mathbf{G_L}$ *is not the perfect matching graph* **do**
  $S \leftarrow$ the free node in X;
  $H =$ hungarian trees of S;
  $T = \{c_{fj}^1 | e_{ij} = (c_{fi}^0, c_{fj}^1) \in \mathbf{G_L} \bigwedge c_{fi}^0 \notin H\}$;
  $S = S \cup (X \cap H)$;
  for $e_{i,j}$ *is not an augmenting path* **do**
    $S \leftarrow c_{fi'}^0$, $T \leftarrow c_{fj'}^1$, $\mathbf{G_L} \leftarrow e_{i',j'}$;
  end
end
$\mathbf{G_D} = \{e_{ij} | e_{ij} \in \mathbf{G_L} \wedge e_{ij} < \theta\}$;
$\mathbf{G_L} = \mathbf{G_L} - \mathbf{G_D}$;

called privacy amplification [17]. Note that to preserve the confidentiality of the key, privacy amplification will remove some bits from the key after each round of parity check. Hence in key generation, Alice can reserve some additional bits for information reconciliation. For example, she will generate 300 bits for a 256-bit secret key. Alice will terminate information reconciliation after a specified number of rounds. In fact, in our experiments, the bit error rate is very low (< 0.01 for two devices within 4cm distance in outdoor environments). Hence the cost of information reconciliation is low.

4. ANALYSIS AND DISCUSSION

In this section, we discuss and analyze the security and efficiency of TDS.

4.1 Security of TDS

The authenticity, confidentiality, and integrity of TDS can be guaranteed under the framework of information reconciliation [3] [22]. In the cascade protocol [3], both Alice and Bob have a version of a key and the two versions contain mismatched bits. They use parity checking via a public channel to correct the errors. The model completely describes our protocol. It has been shown that information reconciliation is essentially source coding with side information. The amount of information to be exchanged in optimal information reconciliation is the conditional Shannon entropy and information reconciliation and privacy amplification are information-theoretically secure [22] [17]. We present the authenticity and confidentiality protection in other steps than information reconciliation.

Authenticity. Eve, located out of the safe distance from Alice, may want to pretend to be a legitimate device and run information reconciliation. The authenticity is protected because Alice will only run information reconciliation for a fixed number of rounds for every other device. Hence only if the bit error rate is smaller than a reasonable threshold, *e.g.*, 7%, Eve can get the key obtained by Alice. According to results in Section 5.3, the bit error rate of any device with $> 12cm$ distance from Alice is around 50%, the maximum bit error rate. In addition, any bit exposed during information reconciliation will be removed from the key. Hence Eve cannot perform spoofing.

Confidentiality. In addition to the above framework, we need to demonstrate that the singular values broadcast by Alice reveal no information about the secret bits. As shown in Figure 8, the singular values of 0-blocks and 1-blocks have identical distributions. Hence given two singular values, the eavesdropper still cannot improve its guess on this bit.

4.2 Predictable channel attack

A significant concern about reciprocal quantization is that an adversary can use deliberately planned movements to generate desired or predictable changes in the channel between the legitimate devices. Unfortunately prior works cannot defend against such a predictable channel attack [8].

TDS does not use reciprocal quantization. The key of TDS is generated by Alice using sophisticated algorithms. Even if the adversary performs deliberate actions, *e.g.*, interfere the channel, it cannot yield any predictable patten on generated key bits. We should guarantee that the key delivery process is also resilient to the predictable channel attacks. TDS uses S-box for key delivery, in which the features used to represent secrete information should be unpredictable. As we discussed above, the features representing 0s and 1s are independent and identically distributed. After block allocation, the measurements have been sufficiently diffused and confused, as to meet the Shannon's diffusion and confusion properties in conventional cryptography ciphers. In this case, the adversary cannot generate a predictable pattern over the measurements in TDS's blocks, even if she is able to manipulate predictable patterns in the channel by deliberate actions, such as blocking the channel periodically. Therefore, TDS can effectively defend against predictable channel attacks. We will show our experimental study of this point in Subsection 5.5.

4.3 Stability of KM based feature paring algorithm

In Subsection 3.3, we have discussed the KM based feature pairing algorithm, which can generate a maximum matching graph.

In Algorithm 1, we discard all pairs whose difference is less than θ, which is about 5%. In this section, we will demonstrate that KM algorithm is stable, *i.e.,* the remaining graph excluding minimal edge is also a maximum matching graph. We have no need to re-carry the Algorithm 1. Denote the maximum weight of graph G before discarding to be M, and the maximum weight after discarding edge $e_{i_0j_0}$, named G', to be $M - w(e_{i_0j_0})$. Assuming that the maximum weight of G' is M', we have $M' > M - w(e_{i_0j_0}) \Rightarrow M < M' + w(e_{i_0j_0})$, *i.e.*, there exists another matching weigh $M' + w(e_{i_0j_0})$ of G is larger than M. This result derived from that assumption, obviously, is inconsistent with the fact that M is the maximum weight of G. Hence, that assumption is false and Algorithm 1 is stable.

4.4 Fault tolerance

Due to the presence of noises and manufacture variations, there may be a difference of CSI measurements h_i in the i_{th} sample, denoted as δ_i. When δ is larger than ε, $\Delta\hat{\sigma}$ begins to incur mismatched bits, which leads to a wrong information delivery. Using multiple samples in a block can reduce the variance of the represented features. According to Chebyshev inequality, we have $P\{|\delta - E(\delta)| \geq \varepsilon\} \leq \frac{D(\delta)}{\varepsilon^2}$. Block-based information delivery can efficiently reduce the variance of average δ, and then reduce the secret bit error rate.

TDS extracts the feature of block based on SVD. As aforementioned in Section 3.2, the block size is $10 \times n$ (typically $n = 6$). SVD can be expressed as $G = U\hat{\Sigma}V^T = \sum_{i=1}^{\beta} \hat{\sigma_i} U_i V_i^T$, where $\hat{\sigma}$ is the singular value of G, and U_i, V_i are the i_{th} column vectors of U and V, respectively. The power of noise is $P_N = \sum_1^{\beta} (\sigma_i^w)^2$, where σ_i^w is the i_{th} singular value of noise matrix. TDS uses the second or third singular values $\hat{\sigma}_2$ and $\hat{\sigma}_3$ to represent the signal features and discards the singular value smaller than $\hat{\sigma}_4$ which are mainly relevant to noises. Therefore, the noise is decreased by $\sum_{i=4}^{\beta} (\sigma_i^w)^2$ through SVD.

4.5 Information delivery rate

We use the number of delivered secret bits per sample as the information delivery rate. In order to further improve the information delivery rate, TDS can divide one block into two orthogonal sets of samples to transfer two bits. In Figure 6, we find that for a given subcarrier, the correlations between it and other subcarriers vary gradually, and there should be another subcarrier with the lowest correlation coefficient, *i.e.* most uncorrelated to it. For example, the correlation between the 1st subcarrier and the 13th subcarrier is almost zero. TDS divides all subcarriers two sets: H_1: ($\{1, 2, \cdots, 10\}$, $\{1, 3, \cdots, 19\}$, and $\{1, 4, \cdots, 28\}$), and H_2: ($\{13, 14, \cdots, 22\}$, $\{13, 15, \cdots, 29, 1\}$, and $\{14, 17, \cdots, 29, 2, \cdots, 11\}$). The distributions of $\Delta\hat{\sigma}$ in the two sets are plotted in Figure 9. The blue and red points are the $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ distributions in H_1, while the black and green points are the distributions in H_2. These two sets are independent. Their $\Delta\hat{\sigma}$ can be used to deliver two bits in one block. Therefore, the delivery rate of TDS can be doubled. In our system, we set $n = 4$ in mobile scenarios and $n = 6$ in static scenarios, and their delivery rate is $\frac{6}{4}$ and $\frac{4}{6}$. That is, each sample can confidentially deliver 1.5 bits and 0.67 bit in mobile scenarios and static scenarios, respectively.

4.6 Distance constraint and multi-user key agreement

To agree on a shared secret key with reliability, all devices need to be located close to each other. Therefore, the authentication distance will impact the scalability of TDS. TDS allows a key to be directly delivered from one device to others, saving huge amount of overhead from interactive agreement. The requirement is that all devices are located in the close proximity centered at the sender. When the number of legitimate devices increases, their distance between each other may also increase due to space limit, which will reduce the consistency of channel measurements due to fast fading channel.

Table 1: Experiments scenarios

Index	State	Environment
A	Static	Indoor
B	Static	Outdoor
C	Mobile	Indoor
D	Mobile	Outdoor

Table 2: NIST statistical test results. To pass this test, p-value must be greater than 0.01.

Test	A	B	C	D
Monobit Frequency	0.611	0.757	0.900	0.784
Longest Run of Ones	0.724	0.660	0.861	0.883
FFT	0.553	0.848	0.757	0.752
Approximate Entropy	0.708	0.897	0.899	0.719
Cumulative Sums (Fwd)	0.530	0.776	0.905	0.681
Cumulative Sums (Rev)	0.787	0.749	0.955	0.919
Block Frequency	0.725	0.819	0.874	0.977
Runs	0.734	0.723	0.883	0.846
Serial	0.421	0.401	0.841	0.885
	0.590	0.530	0.913	0.642

In order to make TDS work well for many devices, *e.g.*, more than three, we propose a new communication model to beyond the space limitation. Instead of using a public wireless source, Alice and Bob ping each other to generate symmetric random channel variations. Other legitimate devices are located near Alice and Bob within authentication distance to hear the communication between Alice and Bob. This model can double the authentication space to support key agreement for more users.

5. IMPLEMENTATION AND EVALUATION

In this section, we present the prototype implementation, experiment setup, and performance evaluation of TDS.

5.1 Methodology

We conduct extensive experiments with five laptop computers, named Alice, Bob, Calvin, Eve, and Peter. The laptops are all equipped with commodity off-the-shelf wireless NICs model Intel 5300. Peter is configured as an AP. The wireless connection among five laptops operates in the 802.11n 2.4GHz channel. Antennas of Alice, Bob and Calvin are located in less than 5cm (0.4λ) distance, while Eve is deployed at least 25cm (2λ) away from Alice. As the AP, Peter broadcasts beacons every 50ms. In two users mode, Alice pings Bob every 50ms and receives Bob's ACK after 1-5ms. Alice broadcasts Timing Synchronization Function (TSF) timestamp to synchronize all legitimate devices within 25 microseconds. Eve turns itself into the monitor mode to be an eavesdropper.

We conduct our experiments in a large variety of environmental settings and under different scenarios as listed in Table 1. In our experiments in static environments (A and B), there is no line of sight between Alice and Bob, and all the objects are keeping still. In the other experiments in mobile environments (C and D), with several intermediate

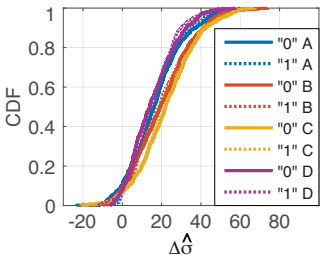


Figure 11: Distribution of $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$

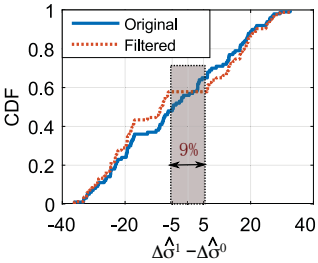


Figure 12: Distribution of differences between $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$

objects, the presence or the absence of line of sight changes with time. In different scenarios, we use the following metrics for performance evaluation:

Bit generation rate is defined as the number of secure bits of the key over the overall time for key agreement. Note the time is for the entire process including CSI measurement, S-box construction, and key delivery.

Bit error rate is the number of mismatched bits over the number of all bits generated.

Randomness and entropy is used to evaluate the quality of keys. We measure the randomness of the keys generated by TDS using the standard NIST test. We also compute the entropy of the key generated.

All results are the average value from at least 20 independent experiments.

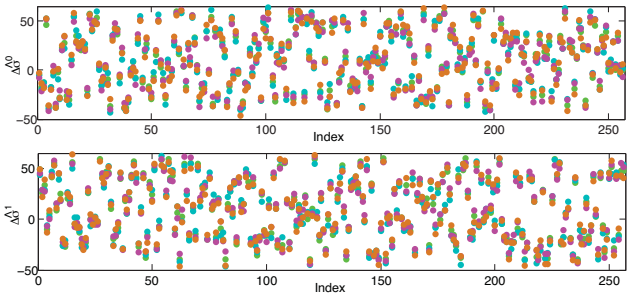


Figure 13: The distribution of $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ for multiple users.

5.2 Randomness of key and block features

Since we assume Eve has complete information of the protocol, any non-random behavior in the bit sequences or block features can be exploited by the adversary to reduce the time-complexity of cracking the key.

Although Alice can generate an arbitrary key, the key may also be slightly changed after privacy amplification. We employ a widely used randomness test suit, NIST to verify the randomness of the secret-bit generated by TDS. In this test, we use 200 bit sequences generated from our experiments in scenarios A, B, C, and D, and compute their p-values for 8 types of tests. According to the specification in this suite, if all p-values are greater than 0.05, the sequence is random. We list the p-values of TDS in Table 2. From the results, we find that the bit streams generated by TDS pass all the tests with high values.

TDS uses $\Delta\hat{\sigma}$ to transmit secret information, $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ is the feature of CSI measurements in one block used to represent 0 and 1 respectively. They should have independent and identical distribution to avoid information leakage. Figure 11 shows the distribution of $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ in scenarios A, B, C, and D. The distributions in different scenarios are slightly different. In the same scenario, $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ have extremely similar distributions. Therefore, the adversary can hardly obtain any information from the $\Delta\hat{\sigma}$ delivered in public wireless channels.

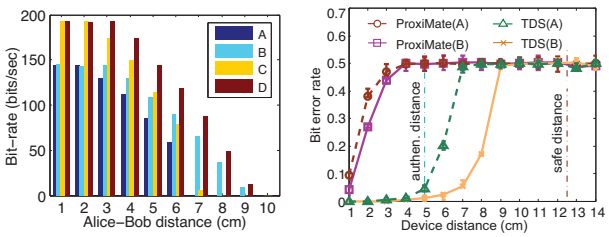
In addition, the differences between $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ for each 0/1 pair influence bit error rate. A large differentiation of "0/1" for each bit will enhance the fault tolerance. We take a paring algorithm based on Max-Weighted Bipartite Matching to solve this problem. Figure 12 shows the distribution of the differences between $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ for each 0/1 pair. The differences for original 0/1 pair are nearly a linear distribution. There are about 9% pairs with the differences less than 10. We introduce a filtered perfect matching method to filter the pairs with small differentiation.

5.3 How distance influences performance

Figure 14(a) shows the bit generation rate by varying the distance between two devices (Alice and another receiver of the key). We find that when the distance is smaller than 4cm, the bit generation rate is always higher than 100 bits/sec. Hence it only takes a couple of seconds to get a 256-bit key. The bit generation rate in mobile scenarios is higher than that in static scenarios. The bit generation rate in outdoor environments is higher than that in indoor environments. It is because mobile and outdoor environments provide more channel diversity. Compared with another method ProxiMate [15] that only generate a few bits per second, the bit generation rate of TDS is higher by more than an order of magnitude.

Figure 14(b) shows the bit error rate by varying the distance between devices, for ProxiMate and TDS. Even if the distance of two device antennas is 1cm, the bit error rate of ProxiMate is about 5%-10%. For TDS, when the distance is less than 3cm, the mismatch rate of TDS is 0 for outdoor environments and < 0.015 for indoor environments. When the distance is 5cm, the mismatch rate of TDS is still smaller than 7%. We mark the authenticate distance and safe distance in the figure. Here the safe distance can be set to 12.5cm but a user can easily check a much longer safe distance such as 25cm or even 50cm. Out side of the safe distance, a device has bit error rate equal to 0.5, the maximum bit error rate.

Figure 14(c) shows the parity check counts with increasing the distance between devices, for ProxiMate and TDS. The number of passes is 5. When the distance is more than 1cm, parity check counts of ProxiMate are larger than 130, which might not work properly. For TDS, as long as the distance is less than 5cm, the parity check counts are less than 20 in both indoor and outdoor scenarios. The devices within 5cm can achieve pairing without user intervention. For large civilian or military transceivers, we may use external antennas which can be easily placed in 5cm.



(a) Bit generation rate of TDS

(b) Bit error rate

(c) Rounds of parity checks

Figure 14: Key generation performance vs. distance for TDS & ProxiMate

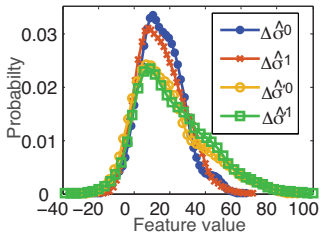


Figure 15: Distributions of the features without and under attacks

5.4 Group key agreement

TDS supports group key agreement. For the situation with more than two devices, devices adopt the new communication model to deal with fast fading channel proposed in Section 4.6. Figure 13 plots the distribution of $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ for 4 users. Four colors of points represent four devices. The points in the same column are $\Delta\hat{\sigma}^0$ or $\Delta\hat{\sigma}^1$ of four devices for the same 0/1 pair. Two devices Alice and Bob are 30cm away from each other. Alice pings Bob every l00ms and receives Bob's ACK after 1-5ms. Calvin and Peter are near to Alice and Bob within 4cm respectively. The $\Delta\hat{\sigma}^0$ and $\Delta\hat{\sigma}^1$ of four users are almost identical for the same bit, which can be used to represent secret bits reliably among the group.

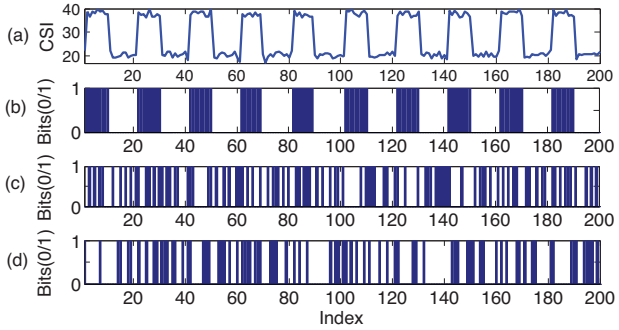


Figure 16: CSI measurements when an intermediate object moving between Alice and Bob.

5.5 Robust against predictable channel attack

The attacker Eve can perform some deliberately planned movements to block the LOS between Alice and Bob, such that the bits extracted from the CSI measurements with manipulated changes, in hope that it can predict the features for 0 and 1 bits as well as the key.

Figure 16(a) shows that CSI measurements from the 1st subcarrier display periodical changes under predictable channel attacks. The CSI values increase when Eve blocks LOS or decrease when Eve moves away. Figure 16(b), (c) and (d) plot the bits of the agreed key by reciprocal quantization, TDS, and KEEP, respectively. The blue parts and white parts represent "1" and "0".

For reciprocal quantization, the generated bits present an predictable pattern. When the channel is blocked, the bits are generated as 0s, otherwise, they are 1s. In contrast, the variations of extracted bits by KEEP and TDS are independent of the blocking pattern. It is because TDS do not rely on the channel condition to generate keys and KEEP extracts keys by randomly picking up discrete fragments from all the subcarriers of OFDM. Therefore, an attacker cannot infer the pattern of the secret bits by TDS based on his interference pattern.

In addition, we should guarantee that the feature distributions are also resilient to the predictable channel attacks. Figure 15 compares the distribution of delivering value with/without predictable channel attack (denoted by $\Delta\hat{\sigma}'$ and $\Delta\hat{\sigma}$, respectively). It reveals that the features represented 0s and 1s are almost identically distributed regardless of predictable channel attack. This is because that the block allocation sufficiently diffuse and confuse the CSI measurements. In this case, the adversary cannot generate a predictable pattern over those measurements, even if it is able to manipulate predictable patterns in the channel by deliberate actions, such as blocking the channel periodically. More interesting, the distribution range of $\Delta\hat{\sigma}'$ grows wider, since the predictable channel attack introduce more variance of measurements. It increases the difference between $\Delta\hat{\sigma}'^0$ and $\Delta\hat{\sigma}'^1$ for each bit, which improves fault-tolerance of S-box. Therefore, TDS can effectively defend against predictable channel attacks.

5.6 Comparison of key extraction approaches

We compare TDS with existing key generation and agreement approaches for mobile networks, including KEEP [28], Mathur *et al.* [16], ASBG [8], CGC [12]. Note these solutions assume an authenticated channel between two devices. Hence they are weaker in security than TDS. We align the baseline of comparison as follows. In the scheme proposed by Mathur *et al.*, there are two parameters α and m. We set $\alpha = 0.35$ and $m = 2$ to ensure most fractions of measurements are used for bit extraction. For ASBG, CGC, and KEEP, we choose $\alpha = 0.35$ and fragment size is 50, where the mismatch ratio is low. For TDS, we choose block size $\beta = 6$ in static scenarios and $\beta = 4$ in mobile scenarios. The distance between Alice and another device is within 4cm.

We compare the entropy of keys generated by different approaches in Figure 17. The entropy can reflect the randomness of keys from the perspective of uncertainty. TDS and KEEP have the highest entropy in all methods, and CGC has the lowest. Figure 18 shows the bit error rates. In this

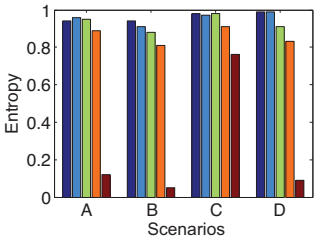


Figure 17: Entropy of the keys

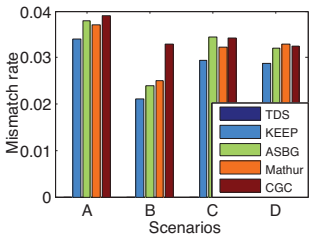


Figure 18: Bit error rate

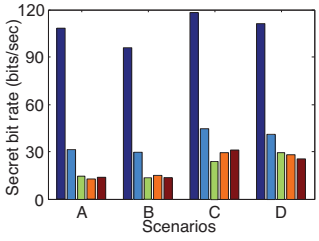


Figure 19: Secret bit generation rate

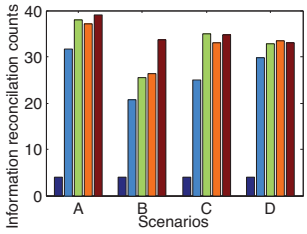


Figure 20: Information reconciliation counts

distance, TDS has no mismatched bit, while other methods may cause around 2% to 4% mismatched bits. Figure 19 shows the bit generation rates. Obviously TDS has significantly higher generation speed. Note the bit generation rate of TDS is slower than previous results. It is because in this set of experiments, Alice and Bob do not listen to a public WiFi but use the communication among them for sampling. This is the only model that the other protocols can work but TDS is not restricted to it. Figure 20 shows the number of rounds for information reconciliation. Since there are no mismatched bit, TDS only uses 4 times pass check to guarantee the consistency of transmitted secret bits.

6. RELATED WORK

To ensure data confidentiality, creating keys based on the physical layer information of wireless channels is promising due to its efficiency and security [18] [25] [5] [7] [26]. Most of existing methods focus on pairwise key generation by measuring the time-varying channel [11] [29] [21]. Exploiting temporal and spatial variations of wireless channels, RSS based techniques are widely used [16][20][8]. They tend to transform the RSS values to a sequence of bits, and create secrets based on the reconciled bits. However, RSS may vary at different receivers, so the key generation rate of RSS based methods is low. For example, Radiotelepathy [16] extracts secret keys using the channel impulse response (CIR) in the wireless channel and its key generation rate is only around 1 bit per second. Pinpoint [27] can fast exchange information exploiting CIR with reversed jamming noise between two devices, yet with little scalability.

Contrast with RSS, CSI is much richer source of secret information. It can be obtained via the Orthogonal Frequency-Division Multiplexing (OFDM). Liu *et al.* [14] theoretically prove the feasibility of CSI and high key generation using CSI. A practical CSI based key exaction system [12] has been implemented which works in both static and mobile environments. However, CSI measurements among adjacent subcarriers have strong correlations, so the key generated from nearby subcarriers also have correlation, which is vulnerable to key cracking attacks. To avoid such a risk, KEEP [28] introduces a validation-recombination mechanism that combines the information of all subcarriers and is resilient to the key cracking attack.

In many applications, it is necessary to establish a collaborative key among a group of wireless devices. Key establishment concerning the shared group key is discussed in [13]. In a group key establishment scheme, each node keeps a matrix, which includes the values measured from all its channels to its neighbors.

In summary, none of existing methods can achieve instant and robust key agreement among multiple devices.

7. CONCLUSION

TDS is a device authentication and key agreement protocol that helps multiple devices to agree on a secret key in a couple of seconds. Compared with prior solutions for mobile networks, it has four important advantages: i) its key generation rate is faster by more than an order of magnitude; ii) it supports more than two devices; iii) it can agree on an arbitrary key with strong randomness; iv) it can effectively defend against predictable channel attacks. We conduct rigorous analysis to show the feasibility and security of our protocol. We also implement TDS in commodity off-the-shelf WiFi devices. The experiment results demonstrate the high efficiency and robustness of TDS. We believe the idea of TDS can be extended in other communication scenarios.

8. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China under Grant No. 61325013, 61190112, 61572396, and 61402359.

Chen Qian is supported by UC Santa Cruz Startup Grant and National Science Foundation grant CNS-1464335.

Sheng Zhong is supported in part by the Jiangsu Province Double Innovation Talent Program and in part by the National Natural Science Foundation of China under Grant No. 61300235, 61321491, 61402223, and 61425024.

Xiang-Yang Li is partially supported by NSF ECCS-1247944, NSF CMMI 1436786, NSF CNS 1526638, National Natural Science Foundation of China under Grant No. 61520106007.

9. REFERENCES

[1] K. Argyraki, S. Diggavi, M. Duarte, C. Fragouli, M. Gatzianas, and P. Kostopoulos. Creating secrets out of erasures. In *Proceedings of ACM MobiCom*, 2013.

[2] B. Azimi-Sadjadi, A. Kiayias, A. Mercado, and B. Yener. Robust key generation from signal envelopes in wireless networks. In *Proceedings of ACM CCS*, 2007.

[3] G. Brassard and L. Salvail. Secret-key reconciliation by public discussion. In *Proceedings of Advances in Cryptology-EUROCRYPT*, 1994.

[4] N. Cheng, X. Oscar Wang, W. Cheng, P. Mohapatra, and A. Seneviratne. Characterizing privacy leakage of public WiFi networks for users on travel. In *Proceedings of IEEE INFOCOM*, pages 2769–2777. IEEE, 2013.

[5] J. Croft, N. Patwari, and S. K. Kasera. Robust uncorrelated bit extraction methodologies for wireless sensors. In *Proceedings of ACM/IEEE IPSN*, pages 70–81. ACM, 2010.

[6] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 1976.

[7] S. Gollakota and D. Katabi. Physical layer wireless security made fast and channel independent. In *Proceedings IEEE of INFOCOM*, pages 1125–1133. IEEE, 2011.

[8] 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 *Proceedings of ACM MobiCom*, 2009.

[9] D. Kalman. A singularly valuable decomposition: the SVD of a matrix. *The college mathematics journal*, 27(1):2–23, 1996.

[10] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics*, 52(1):7–21, 2005.

[11] L. Lai, Y. Liang, and H. V. Poor. A unified framework for key agreement over wireless fading channels. *IEEE Transactions on Information Forensics and Security*, 7(2):480–490, 2012.

[12] H. Liu, Y. Wang, J. Yang, and Y. Chen. Fast and practical secret key extraction by exploiting channel response. In *Proceedings of IEEE INFOCOM*, 2013.

[13] H. Liu, J. Yang, Y. Wang, and Y. Chen. Collaborative secret key extraction leveraging received signal strength in mobile wireless networks. In *Proceedings of IEEE INFOCOM*, pages 927–935. IEEE, 2012.

[14] Y. Liu, S. C. Draper, and A. M. Sayeed. Exploiting channel diversity in secret key generation from multipath fading randomness. *IEEE Transactions on Information Forensics and Security*, 7(5):1484–1497, 2012.

[15] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam. Proximate: proximity-based secure pairing using ambient wireless signals. In *Proceedings of ACM MobiSys*, pages 211–224. ACM, 2011.

[16] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Proceedings of ACM MobiCom*, pages 128–139. ACM, 2008.

[17] U. M. Maurer and S. Wolf. Secret-Key Agreement Over Unauthenticated Public Channelsą‡Part III: Privacy Amplification. *IEEE Transactions on Information Theory*, 2003.

[18] R. Mehmood, J. W. Wallace, M. Jensen, et al. Key establishment employing reconfigurable antennas: Impact of antenna complexity. *IEEE Transactions on Wireless Communications*, 13(11):6300–6310, 2014.

[19] M. Miettinen, N. Asokan, T. D. Nguyen, A. Sadeghi, and M. Sobhani. Context-based zero-interaction pairing and key evolution for advanced personal devices. In *Proceedings of ACM CCS*. ACM, 2014.

[20] N. Patwari, J. Croft, S. Jana, and S. K. Kasera. High-rate uncorrelated bit extraction for shared secret key generation from channel measurements. *IEEE Transactions on Mobile Computing*, 9(1):17–30, 2010.

[21] K. Ren, H. Su, and Q. Wang. Secret key generation exploiting channel characteristics in wireless communications. *IEEE Wireless Communications*, 18(4):6–12, 2011.

[22] R. Renner and S. Wolf. Simple and tight bounds for information reconciliation and privacy amplification. In *Proceedings of ASIACRYPT*, 2005.

[23] Z. Sun, A. Purohit, R. Bose, and P. Zheng. Spartacus: Spatially-aware interaction for mobile devices through energy-efficient audio sensing. In *Proceedings of ACM MobiSys*, 2013.

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

[25] Q. Wang, H. Su, K. Ren, and K. Kim. Fast and scalable secret key generation exploiting channel phase randomness in wireless networks. In *Proceedings of IEEE INFOCOM*, pages 1422–1430. IEEE, 2011.

[26] Q. Wang, K. Xu, and K. Ren. Cooperative Secret Key Generation from Phase Estimation in Narrowband Fading Channels. *IEEE Journal on Selected Areas in Communications*, 30(9):1666 – 1674, 2011.

[27] T. Wang, Y. Liu, Q. Pei, and T. Hou. Location-restricted services access control leveraging pinpoint waveforming. In *ACM Sigsac Conference on Computer and Communications Security*, pages 292–303, 2015.

[28] W. Xi, X.-Y. Li, C. Qian, J. Han, S. Tang, J. Zhao, and K. Zhao. KEEP: Fast Secret Key Extraction Protocol for D2D Communication.

[29] S. Xiao, W. Gong, and D. Towsley. Secure wireless communication with dynamic secrets. In *Proceedings of IEEE INFOCOM*, pages 1–9. IEEE, 2010.