Towards Wireless Security without Computational Assumptions

—An Oblivious Transfer Protocol Based on an Unauthenticated Wireless Channel

Zhuo Hao[*†], Sheng Zhong[*] and Li Erran Li[‡]
[*]Department of Computer Science and Engineering, State University of New York at Buffalo
[†]Department of EEIS, University of Science and Technology of China
[‡]Networking Research Lab, Bell Labs, Alcatel-Lucent
zhuohao@buffalo.edu, szhong@buffalo.edu, erranlli@research.bell-labs.com

***Abstract*—Wireless security has been an active research area since the last decade. A lot of studies of wireless security use cryptographic tools, but traditional cryptographic tools are normally based on computational assumptions, which may turn out to be invalid in the future. Consequently, it is very desirable to build cryptographic tools that do not rely on computational assumptions.**

In this paper, we focus on a crucial cryptographic tool, namely 1-out-of-2 oblivious transfer. This tool plays a central role in cryptography because we can build a cryptographic protocol for any polynomial-time computable function using this tool. We present a novel 1-out-of-2 oblivious transfer protocol based on wireless channel characteristics, which does not rely on any computational assumption. We also illustrate the potential broad applications of this protocol by giving two applications, one on private communications and another on privacy preserving password verification. We have fully implemented this protocol on wireless devices and conducted experiments in real environments to evaluate the protocol and its application to private communications. Our experimental results demonstrate that it has reasonable efficiency.

I. Introduction

Wireless security has been an active research area since the last decade. A lot of studies of wireless security use cryptographic tools such as encryption, authentication, and key agreement in order to achieve security protection. These traditional cryptographic tools are very powerful, but most of them have a common weakness—normally, they are based on computational assumptions.

For example, consider one of the most frequently used cryptographic tools, symmetric key encryption. We have a number of very good existing encryption schemes, e.g., AES [17]. However, when we use AES to encrypt a message, we are actually making an implicit assumption: the AES block cipher is a psedorandom permutation. Intuitively, this assumption implies that it is infeasible for an adversary to find the cleartext message from the ciphertext. Nevertheless, the above assumption of pseudorandomness is based on the cryptologists' understanding of the *current* attacks on encryption schemes. It is possible that, in the future (maybe even in the near future), the AES scheme will be broken by newly invented cryptanlysis techniques.

In fact, there was a lesson a few years ago, when cryptologists broke several famous hash functions, including MD5 and SHA-0 [42], [43]. To be more precise, these hash functions had been assumed to be collision-resistant for more than ten years, but cryptologists found that these assumptions are invalid and there are quite efficient algorithms to find collisions of these hash functions. It is worth noting that the above discoveries were made after the hash functions became either national standards or de facto standards. Hence, it will be very desirable if we can remove cryptographic tools' dependence on such computational assumptions.

Of course, removing computational assumptions from the cryptographic tools, and thus from the wireless security systems, is a highly challenging problem. Consequently, in this paper, we do not intend to build a complete wireless security system that does not rely on computational assumptions. In stead, we would like to address a fundamental question as a crucial step towards solving this very challenging problem: Is it at all feasible to build wireless security systems without relying on computational assumptions?

Our answer to the above question is positive. Specifically, we propose that wireless security can be based on the physical channel characteristics rather than computational assumptions, as illustrated by a new type of protocols for key agreement in wireless networks [30], [45], [6], [39], [32], [26], [34].[1] In other words, the wireless channel characteristics can be used not only to achieve key agreement, but also to establish *any* cryptographic tool.

To be more precise, we use wireless channel characteristics to build a crucial cryptographic tool called 1-out-of-2 oblivious transfer. (For simplicity, hereafter we use OT to refer to oblivious transfer, and use OT_1^2 to refer to 1-out-of-2 oblivious transfer.) The reason for choosing to work on OT_1^2 is that

[1] This is not the only way to do cryptographic operations without computational assumptions; quantum communications do not rely on computational assumptions as well. But quantum communications are out of the scope of this paper.

it plays a central role in cryptography. In fact, Kilian [28] has proved that OT_1^2 is "complete", meaning that for any polynomial-time computable function, we can build a cryptographic protocol using OT_1^2 . For example, electronic voting protocols, anonymous communications protocols, digital cash protocols, privacy preserving data mining protocols, etc. can all be built using OT_1^2 . Hence, once we get an OT_1^2 protocol independent of computational assumptions, we can actually use it to establish other cryptographic protocols independent of computational assumptions.

However, it is not easy to construct an OT_1^2 protocol based on wireless channel characteristics. The main idea underlying our work is to employ a novel technique from [14]. We point out that both our channel model and our OT_1^2 protocol are significantly different from those of [14]. Consequently, our use of their technique is non-trivial.

To illustrate the potential wide applications of our work, we give a method of private communications based on our OT_1^2 protocol. Just like traditional symmetric key encryption schemes, this method allows two wireless devices that have a common secret key to communicate with each other privately. Nevertheless, the security of this method depends on wireless channel characteristics, not on computational assumptions.

Another application of our OT_1^2 protocol is privacy preserving password verification. Using the method we present, one wireless device can verify a password from another wireless device in such a way that the password is not revealed to either the former device or any eavesdropper.

In summary, we have the following contributions in this paper.

- We are the first to construct an OT_1^2 protocol based on the physical characteristics of wireless channels. Our OT_1^2 protocol does not rely on any computational assumptions. Given the completeness of OT_1^2 proved by Kilian [28], our work can be considered a crucial step towards building strong wireless security systems without computational assumptions.
- Our OT_1^2 protocol has wide potential applications. In particular, we have given a method of private communications and a method of privacy preserving password verification based on our own OT_1^2 protocol.
- We have *completely* implemented our OT_1^2 protocol on *real, mobile* wireless devices, and evaluated it through extensive experiments. We have also experimentally evaluated our private communications method. Our experimental results demonstrate that our OT_1^2 protocol and its application to private communications both have reasonable efficiency.

The rest of this paper is organized as follows. In Section II, we present technical preliminaries. In Section III, we design and analyze our OT_1^2 protocol. In Sections IV and V, we show the two applications of our OT_1^2 protocol. The implementation and experiments are described in Section VI. After briefly reviewing related work in Section VII, we conclude in Section VIII.

II. Technical Preliminaries

Throughout this paper, we follow the formulation presented in [32], [44]. For completeness, we briefly review the model of wireless channels and the quantization method in [32] and refer readers to [32] for more details. After that, we specify the requirements that an OT_1^2 protocol needs to satisfy and the security model we use to analyze OT_1^2 and its applications.

Model of Wireless Channel Consider two parties A and B, and the wireless channel between them. Just like in [32], for ease of presentation, let h be the magnitude of the in-phase component of the Rayleigh fading process, which follows a Gaussian distribution. (Note that our protocol and analysis do not rely on this assumption of distribution. In fact, they can be easily extended to the general case; but the extension is notationally complex and less easy to understand.) Clearly, h can be viewed as a stochastic process; we use $h(t)$ to represent the value of h at time t.

A and B do not know the precise values of $h(t)$; they can only make estimates. Specifically, let $s(t)$ be a well known probe signal. Suppose that B sends a probe signal and A receives it at time t_1; A sends a probe signal and B receives it at time t_2. Then A and B can estimate the channel respectively, using their received signals. In this case, the signals A and B receive can be expressed as follows:

$$r_a(t_1) = h(t_1)s(t_1) + n_a(t_1), \tag{1}$$

$$r_b(t_2) = h(t_2)s(t_2) + n_b(t_2), \tag{2}$$

where $n_a(t_1)$ and $n_b(t_2)$ are the receiver noises at A and B.

By using existing techniques of channel estimation, e.g., [41], A (resp., B) can obtain an estimate $\hat{h}_a(t_1)$ (resp., $\hat{h}_b(t_2)$) from $r_a(t_1)$ (resp., $r_b(t_2)$). These estimates satisfy the following equations:

$$\hat{h}_a(t_1) = h(t_1) + z_a(t_1), \tag{3}$$

$$\hat{h}_b(t_2) = h(t_2) + z_b(t_2), \tag{4}$$

where $z_a(t_1)$ (resp., $z_b(t_2)$) represents the noise and interferences caused by $n_a(t_1)$ (resp., $n_b(t_2)$) during the process of channel estimation.

By the channel reciprocity[2], we can guarantee that $h(t_1)$ and $h(t_2)$ are correlated, if $t_2 - t_1$ is small in the above probe and estimation process. More precisely, we need that the pair of probe signals exchanged by A and B are within the *coherence time* [37], [32] of the wireless channel. Here the coherence time T_C is typically inversely proportional to the maximum Doppler frequency f_m [37], [32]:

$$T_C \approx \frac{1}{f_m} = \frac{\lambda}{v}. \tag{5}$$

In equation (5), λ is the wavelength of the carrier signal, and v is the maximum moving speed of objects in the environment.

[2] If the involved wireless devices are not calibrated, methods similar to [32] can be used to reduce the problem introduced by the lack of calibration.

Note that the above description refers to the exchange of one single pair of probe signals. As we will see, our OT$^2_1$ protocol actually requires exchanges of multiple pairs of probe signals. Unlike the short time interval between the two probe signals in the same pair, the time interval between any two different pairs of probe signals is chosen to be larger than the coherence time. In this way, the channel estimates derived from different pairs of probe signals can be seen as independent from each other.

Method of Quantization When A and B have obtained their estimates $\hat{h}_a$ and $\hat{h}_b$, respectively, they quantize these channel estimates into bit strings using a quantization function Q. The function Q is defined as follows:

$$Q(x) = \begin{cases} 1 & if\ x > q_+ \\ 0 & if\ x < q_- \end{cases} \tag{6}$$

where q_+ and q_- are derived from the mean and standard deviation of channel estimates. Denote the mean by m and the standard deviation by σ. Let α ($\alpha > 0$) be a system parameter. We have

$$q_{+,-} = m \pm \alpha \cdot \sigma. \tag{7}$$

Requirements for OT$^2_1$ and Security Model Our main objective in this paper is to build an OT$^2_1$ protocol between A and B. In Section III, we describe how to build this protocol, including how to use the method of quantization mentioned above. Before we build the OT$^2_1$ protocol, we need to first list the requirements for OT$^2_1$.

Assume that A has two bits b_0 and b_1 as her input, and that B has a bit s as his input. The requirements of an OT$^2_1$ protocol is that, when the protocol terminates,

1) B gets the bit b_s;
2) B gets no information about b_{1-s};
3) A gets no information about s.

Throughout this paper, we analyze the security of OT$^2_1$ and its applications in the semi-honest model, which is one of the standard security models [18]. In this model, each involved party follows the protocol, but they may be curious in learning private information that they are not supposed to learn. Furthermore, eavesdropping by outsiders (i.e., parties not supposed to participate in the protocol) are allowed in our model.

III. OT$^2_1$ based on Wireless Channel Characteristics

Using the probing, estimation, and quantization process described in Section II, now we design an OT$^2_1$ protocol and analyze it.

A. The OT$^2_1$ Protocol

Our OT$^2_1$ protocol consists of two stages. In the first stage, the two parties send multiple probe signals to each other alternately, estimate the channel, and convert the estimates into bits, using the quantization method described in Section II. (Recall that the time interval between each pair of probe signals is within the coherence time, but the time interval between any two different pairs of probe signals is more than the coherence time.) The two parties terminate the first stage as soon as each of them have obtained at least N bits, where N is an even number and a system parameter.

The main idea of the second stage is that A can xor her two secret bits with two sequences of masks respectively and then send the results to B. In order to guarantee that B gets only b_s but not b_{1-s}, we only need to make sure that the sequence of masks for b_s is known to B, but the other sequence is unknown to B. To achieve this objective, we have the following crucial observation:[3] Consider two pairs of probe signals such that A extracts the same bit from them using the quantization method in Section II. From these two pairs of probe signals, if B also extracts the same bit, then it is very likely that the bit extracted by A is equal to the bit extracted by B. In contrast, if from the two pairs of probe signals B extracts two different bits, then B has no idea about what bit is extracted by A. Consequently, for both sequences of masks, we let A use bits extracted from probe signals by A such that the next extracted bits are the same. In order to ensure the sequence of masks for b_s is known to B, we make sure that the masks for b_s correspond to those bits extracted by B that are identical to their next bits. In order to ensure the sequence of masks for b_{1-s} is unknown to B, we make sure that the masks for b_{1-s} correspond to those bits extracted by B that are not identical to their next bits.

More details of the second stage are given below.

Suppose that, at the end of the first stage, A has obtained N bits from the quantized channel estimates: $\{BS_a(i)\}_{i=1,2,\ldots,N}$; B has also obtained N bits from the quantized channel estimates: $\{BS_b(i)\}_{i=1,2,\ldots,N}$. (Note that we use $BS_a(i)$ to denote the ith term in the sequence BS_a. Similar notations are used throughout the paper.) The second stage can be divided into four steps.

Step 1. A generates an index sequence I by extracting all index i such that $BS_a(2i-1) = BS_a(2i)$ ($i \in [1, N/2]$). A sends I to B using a reliable communication protocol, e.g., TCP. Note that, throughout this OT$^2_1$ protocol, communications using this reliable communication protocol need *not* to be encrypted.

Step 2. After B receives the index sequence I from A, B generates two disjoint index sequences I_s and I_{1-s}, where I_s is subject to the following constraints:

(1) $|I_s| = n$ (n is a security parameter), i.e., there are exactly n indices in the sequence I_s;

(2) $I_s \subseteq I$, i.e., I_s is a subsequence of I;

(3) for all $i \in I_s$, $BS_b(2i-1) = BS_b(2i)$;

and I_{1-s} is subject to the following constraints:

(1) $|I_{1-s}| = n$;

(2) $I_{1-s} \subseteq I$;

[3] This observation is valid under the condition that the time interval between the two pairs of probe signals is more than the coherence time. Recall this condition is satisfied by our OT$^2_1$ protocol.

(3) for all $i \in I_{1-s}$, $BS_b(2i-1) \neq BS_b(2i)$.

Then B sends the two index sequences I_0 and I_1 to A, using a reliable communication protocol.

Step 3. Once A receives I_0 and I_1 from B, A generates two sequences L_0 and L_1 as follows: for each $m \in \{0, 1\}$ and each j such that $1 \leq j \leq n$,

$$L_m(j) = b_m \oplus BS_a(2 \cdot I_m(j)),$$

Then A sends L_0 and L_1 to B using a reliable communication protocol.

Step 4. After B receives the L_0 and L_1 from A, B computes $b_s^{'}$ using the following formula:

$$b_s^{'} = \mathsf{majority}(\{L_s(j) \oplus BS_b(2 \cdot I_s(j)), j \in [1, n]\}).$$

Here b_s' is supposed to be equal to b_s, the value B needs to obtain. (In Section III-B, we prove there is a high probability that $b_s' = b_s$.)

A formal description of the second stage is shown in Algorithm 1.

Algorithm 1: Second Stage of OT_1^2 Protocol

```
Input: $\{BS_a(i)\}_{i=1,2,\ldots,N}$ and $\{BS_b(i)\}_{i=1,2,\ldots,N}$
       $A$'s secret bits $\{b_0, b_1\}$, $B$'s secret bit $s$
Output: $B$ outputs $b_s^{'}$ as an estimate of his chosen $b_s$

$A$:
$I \leftarrow$ empty sequence
foreach $i \in [1, N/2]$ do
    if $BS_a(2i-1) = BS_a(2i)$ then
        add $i$ into $I$
    end
end
$A$ sends $I$ to $B$

$B$:
$I_s \leftarrow$ empty sequence, $I_{1-s} \leftarrow$ empty sequence
foreach $i \in I$ do
    if $BS_b(2i-1) = BS_b(2i)$ and $|I_s| < n$ then
        add $i$ into $I_s$
    end
    else if $BS_b(2i-1) \neq BS_b(2i)$ and $|I_{1-s}| < n$ then
        add $i$ into $I_{1-s}$
    end
    if $|I_s| = n$ and $|I_{1-s}| = n$ then
        break
    end
end
$B$ sends $I_s$ and $I_{1-s}$ to $A$

$A$:
$L_0 \leftarrow$ empty sequence, $L_1 \leftarrow$ empty sequence
foreach $m \in \{0, 1\}$ do
    foreach $j \in [1, n]$ do
        $L_m(j) = b_m \oplus BS_a(2 \cdot I_m(j))$
    end
end
$A$ sends $L_0$ and $L_1$ to $B$.

$B$:
$b_s^{'} = \mathsf{majority}(\{L_s(j) \oplus BS_b(2 \cdot I_s(j)), j \in [1, n]\})$.
```

B. Protocol Analysis

Below we show that the three requirements for OT_1^2 are all satisfied by our protocol.

Theorem 1. *Under the standard assumptions [30], [32], [31] that the stochastic process $\boldsymbol{h}$ is stationary and that $h(t)$ is a Gaussian random variable, when our OT_1^2 protocol is finished, for any $\epsilon > 0$, B gets b_s with probability $1 - \epsilon$ as long as*

$$n \geq \frac{ln(\frac{1}{\epsilon})}{2(q - \frac{1}{2})^2},$$

where for any $i \in I_s$,

$$\begin{aligned}\Pr[BS_b(2i-1, 2i) = BS_a(2i-1, 2i) | BS_a(2i) = BS_a(2i-1)] \\ = q > \tfrac{1}{2}.\end{aligned}$$

Proof: Let $\boldsymbol{x_1} = [BS_b(2i-1), BS_a(2i-1), BS_b(2i), BS_a(2i)]^T$ and $\boldsymbol{x_2} = [BS_a(2i-1), BS_a(2i)]^T$ be two random vectors. Since $\boldsymbol{h}$ is a stationary Gaussian process, $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are both random vectors following multivariate Gaussian distributions. Now we consider the following probability. For each $i \in I_s$, we have:

$$\begin{aligned}
&\Pr[BS_b(2i-1, 2i) = \text{“11”} | BS_a(2i-1, 2i) = \text{“11”}] \\
&= \frac{\Pr[BS_b(2i-1, 2i) = \text{“11”}, BS_a(2i-1, 2i) = \text{“11”}]}{\Pr[BS_a(2i-1, 2i) = \text{“11”}]} \\
&= (\int_{q_+}^{+\infty}\int_{q_+}^{+\infty}\int_{q_+}^{+\infty}\int_{q_+}^{+\infty} \frac{1}{(2\pi)^2 |\mathsf{Cov}_{4,4}(\boldsymbol{x_1})|^{1/2}} \cdot \\
&\qquad \exp\{-\frac{1}{2}(\boldsymbol{x_1} - \boldsymbol{\mu_1})^T \cdot \mathsf{Cov}_{4,4}^{-1}(\boldsymbol{x_1}) \cdot (\boldsymbol{x_1} - \boldsymbol{\mu_1})\} d^{(4)}x) \Big/ \\
&(\int_{q_+}^{+\infty}\int_{q_+}^{+\infty} \frac{1}{(2\pi)|\mathsf{Cov}_{2,2}(\boldsymbol{x_2})|^{1/2}} \cdot \\
&\exp\{-\frac{1}{2}(\boldsymbol{x_2} - \boldsymbol{\mu_2})^T \cdot \mathsf{Cov}_{2,2}^{-1}(\boldsymbol{x_2}) \cdot (\boldsymbol{x_2} - \boldsymbol{\mu_2})\} d^{(2)}x)
\end{aligned} \tag{8}$$

In equation (8), $\boldsymbol{\mu_1}$ and $\boldsymbol{\mu_2}$ are the expectation vectors of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$; $\mathsf{Cov}_{4,4}(\boldsymbol{x_1})$ and $\mathsf{Cov}_{2,2}(\boldsymbol{x_2})$ are the covariance matrices of random vectors $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. Similarly,

$$\begin{aligned}
&\Pr[BS_b(2i-1, 2i) = \text{“00”} | BS_a(2i-1, 2i) = \text{“00”}] \\
&= (\int_{-\infty}^{q_-}\int_{-\infty}^{q_-}\int_{-\infty}^{q_-}\int_{-\infty}^{q_-} \frac{1}{(2\pi)^2 |\mathsf{Cov}_{4,4}(\boldsymbol{x_1})|^{1/2}} \cdot \\
&\qquad \exp\{-\frac{1}{2}(\boldsymbol{x_1} - \boldsymbol{\mu_1})^T \cdot \mathsf{Cov}_{4,4}^{-1}(\boldsymbol{x_1}) \cdot (\boldsymbol{x_1} - \boldsymbol{\mu_1})\} d^{(4)}x) \Big/ \\
&(\int_{-\infty}^{q_-}\int_{-\infty}^{q_-} \frac{1}{(2\pi)|\mathsf{Cov}_{2,2}(\boldsymbol{x_2})|^{1/2}} \cdot \\
&\qquad \exp\{-\frac{1}{2}(\boldsymbol{x_2} - \boldsymbol{\mu_2})^T \cdot \mathsf{Cov}_{2,2}^{-1}(\boldsymbol{x_2}) \cdot (\boldsymbol{x_2} - \boldsymbol{\mu_2})\} d^{(2)}x)
\end{aligned}$$

Since the underlying Gaussian process $\boldsymbol{h}$ is stationary, the Gaussian distributions of both $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are symmetric. And also note that q_+ and q_- are symmetric with the mean as the center, so we can get the following equation:

$$\begin{aligned}&\mathsf{Pr}[BS_b(2i-1,2i)=\text{“00”}|BS_a(2i-1,2i)=\text{“00”}]\\ =&\mathsf{Pr}[BS_b(2i-1,2i)=\text{“11”}|BS_a(2i-1,2i)=\text{“11”}].\end{aligned} \tag{9}$$

On the other hand, for each $i \in I_s$,

$$\begin{aligned}&\mathsf{Pr}[BS_b(2i-1,2i)=BS_a(2i-1,2i)|\\ &\qquad BS_a(2i)=BS_a(2i-1)]\\ =&\mathsf{Pr}[BS_a(2i)=1|BS_a(2i)=BS_a(2i-1)]\cdot\\ &\quad \mathsf{Pr}[BS_b(2i-1,2i)=\text{“11”}|BS_a(2i-1,2i)=\text{“11”}]\\ &+\mathsf{Pr}[BS_a(2i)=0|BS_a(2i)=BS_a(2i-1)]\cdot\\ &\quad \mathsf{Pr}[BS_b(2i-1,2i)=\text{“00”}|BS_a(2i-1,2i)=\text{“00”}].\end{aligned} \tag{10}$$

By combining (9) and (10), we get that

$$\begin{aligned}&\mathsf{Pr}[BS_b(2i-1,2i)=BS_a(2i-1,2i)|\\ &\qquad BS_a(2i)=BS_a(2i-1)]\\ =&\mathsf{Pr}[BS_b(2i-1,2i)=\text{“11”}|BS_a(2i-1,2i)=\text{“11”}]\\ =&\mathsf{Pr}[BS_b(2i-1,2i)=\text{“00”}|BS_a(2i-1,2i)=\text{“00”}].\end{aligned}$$

Recall $\mathsf{Pr}[BS_b(2i-1,2i) = BS_a(2i-1,2i)|BS_a(2i) = BS_a(2i-1)] = q$ for any $i \in I_s$. From the way I_s is generated, we know that

$$\forall i \in I_s, BS_a(2i)=BS_a(2i-1), BS_b(2i)=BS_b(2i-1).$$

So, for any $i \in I_s$, $\mathsf{Pr}[BS_b(2i) = BS_a(2i)] = q$. We can rewrite it as

$$\mathsf{Pr}[BS_a(2\cdot I_s(j))=BS_b(2\cdot I_s(j))]=q,$$

where $j \in [1,n]$. The probability that B gets b_s is

$$\begin{aligned}&\mathsf{Pr}[b_s=b_s^{'}]\\ =&\mathsf{Pr}[b_s=\mathsf{majority}(\{L_s(j)\oplus BS_b(2\cdot I_s(j)), j\in[1,n]\})]\\ =&\mathsf{Pr}[b_s=\mathsf{majority}(\{b_s\oplus BS_a(2\cdot I_s(j))\oplus BS_b(2\cdot I_s(j)),\\ &\qquad j\in[1,n]\})]\\ =&\mathsf{Pr}[|\{BS_a(2\cdot I_s(j))=BS_b(2\cdot I_s(j)), j\in[1,n]\}|>\frac{n}{2}],\end{aligned}$$

Because the time interval between any two different pairs of probe signals are greater than the coherence time, the n events $\{BS_a(2 \cdot I_s(j)) = BS_b(2 \cdot I_s(j))\}$, $j \in [1,n]$ are all independent. For each $j \in [1,n]$, define an indicator random variable

$$Ind_j=\begin{cases}1, \text{ if } & BS_a(2\cdot I_s(j))=BS_b(2\cdot I_s(j)),\\ 0, \text{ if } & BS_a(2\cdot I_s(j))\neq BS_b(2\cdot I_s(j)).\end{cases}$$

Then $Ind_1, Ind_2, ..., Ind_n$ are a sequence of independent Bernoulli random variables [38] with parameter q. Let $X(n,q) = |\{Ind_j = 1, j \in [1,n]\}|$. Then $X(n,q)$ is a random variable following the binomial distribution $Binomial(n,q)$. Therefore,

$$\begin{aligned}&\mathsf{Pr}[b_s=b_s^{'}]\\ =&\mathsf{Pr}[|\{BS_a(2\cdot I_s(j))=BS_b(2\cdot I_s(j)), j\in[1,n]\}|>\frac{n}{2}]\\ =&\mathsf{Pr}[|\{Ind_j=1, j\in[1,n]\}|>\frac{n}{2}]=\mathsf{Pr}[X(n,q)>\frac{n}{2}]\\ =&\sum_{i=\lfloor\frac{n}{2}\rfloor+1}^{n}\binom{n}{i}q^i(1-q)^{(n-i)}=1-\sum_{i=0}^{\lfloor\frac{n}{2}\rfloor}\binom{n}{i}q^i(1-q)^{(n-i)}.\end{aligned}$$

Using the Hoeffding inequality [23], we can bound the above probability as follows:

$$\begin{aligned}&\mathsf{Pr}[b_s=b_s^{'}]=1-\sum_{i=0}^{\lfloor\frac{n}{2}\rfloor}\binom{n}{i}q^i(1-q)^{(n-i)}\\ &\geq 1-\mathsf{exp}(-2\cdot\frac{(nq-\frac{n}{2})^2}{n})=1-\mathsf{exp}(-2n\cdot(q-\frac{1}{2})^2).\end{aligned} \tag{11}$$

Because $q > \frac{1}{2}$, we can always make $\mathsf{Pr}[b_s = b_s^{'}]$ sufficiently close to 1 by increasing n. In particular, if we want the propability to be not less than $1-\epsilon$, then we can only need to gurantee that $n \geq \frac{ln(\frac{1}{\epsilon})}{2(q-\frac{1}{2})^2}$. ■

Remark. *In Theorem 1 we have assumed $q > \frac{1}{2}$. We stress this is a realistic assumption because q can be controlled by adjusting α.*

Theorem 2. *When our OT$_1^2$ protocol is finished, B gets no information about b_{1-s}.*

Proof: (Sketch) Let's consider the index sequence I_{1-s}. For each $i \in I_{1-s}$, we have that

$$\begin{aligned}&\mathsf{Pr}[BS_a(2i-1,2i)=\text{“00”}|BS_b(2i-1,2i)=\text{“01”}]\\ =&\frac{\mathsf{Pr}[BS_a(2i-1,2i)=\text{“00”}, BS_b(2i-1,2i)=\text{“01”}]}{\mathsf{Pr}[BS_b(2i-1,2i)=\text{“01”}]}\\ =&(\int_{-\infty}^{q_-}\int_{-\infty}^{q_-}\int_{q_+}^{+\infty}\int_{-\infty}^{q_-}\frac{1}{(2\pi)^2|\mathsf{Cov}_{4,4}(\boldsymbol{x_1})|^{1/2}}\cdot\\ &\mathsf{exp}\{-\frac{1}{2}(\boldsymbol{x_1}-\boldsymbol{\mu_1})^T\cdot\mathsf{Cov}_{4,4}^{-1}(\boldsymbol{x_1})\cdot(\boldsymbol{x_1}-\boldsymbol{\mu_1})\}d^{(4)}x)\Big/\\ &\qquad\mathsf{Pr}[BS_b(2i-1,2i)=\text{“01”}]\end{aligned} \tag{12}$$

Using the symmetry property of Gaussian distribution, we get that

$$\begin{aligned}&(\int_{-\infty}^{q_-}\int_{-\infty}^{q_-}\int_{q_+}^{+\infty}\int_{-\infty}^{q_-}\frac{1}{(2\pi)^2|\mathsf{Cov}_{4,4}(\boldsymbol{x_1})|^{1/2}}\cdot\\ &\mathsf{exp}\{-\frac{1}{2}(\boldsymbol{x_1}-\boldsymbol{\mu_1})^T\cdot\mathsf{Cov}_{4,4}^{-1}(\boldsymbol{x_1})\cdot(\boldsymbol{x_1}-\boldsymbol{\mu_1})\}d^{(4)}x)\Big/\\ &\qquad\mathsf{Pr}[BS_b(2i-1,2i)=\text{“01”}]\\ =&(\int_{-\infty}^{q_-}\int_{q_+}^{+\infty}\int_{q_+}^{+\infty}\int_{q_+}^{+\infty}\frac{1}{(2\pi)^2|\mathsf{Cov}_{4,4}(\boldsymbol{x_1})|^{1/2}}\cdot\\ &\mathsf{exp}\{-\frac{1}{2}(\boldsymbol{x_1}-\boldsymbol{\mu_1})^T\cdot\mathsf{Cov}_{4,4}^{-1}(\boldsymbol{x_1})\cdot(\boldsymbol{x_1}-\boldsymbol{\mu_1})\}d^{(4)}x)\Big/\\ &\qquad\mathsf{Pr}[BS_b(2i-1,2i)=\text{“01”}]\end{aligned}$$

$$=\frac{\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”}, BS_b(2i-1,2i) = \text{“01”}]}{\mathsf{Pr}[BS_b(2i-1,2i) = \text{“01”}]}.$$

So,

$$\begin{aligned}&\mathsf{Pr}[BS_a(2i-1,2i) = \text{“00”} | BS_b(2i-1,2i) = \text{“01”}]\\ =&\frac{\mathsf{Pr}[BS_a(2i-1,2i) = \text{“00”}, BS_b(2i-1,2i) = \text{“01”}]}{\mathsf{Pr}[BS_b(2i-1,2i) = \text{“01”}]}\\ =&\frac{\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”}, BS_b(2i-1,2i) = \text{“01”}]}{\mathsf{Pr}[BS_b(2i-1,2i) = \text{“01”}]}\\ =&\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”} | BS_b(2i-1,2i) = \text{“01”}].\end{aligned} \tag{13}$$

Since for each $i \in I_{1-s}$,

$$\begin{aligned}&\mathsf{Pr}[BS_a(2i-1,2i) = \text{“00”} | BS_b(2i-1,2i) = \text{“01”}]+\\ &\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”} | BS_b(2i-1,2i) = \text{“01”}] = 1\end{aligned}$$

we have

$$\begin{aligned}&\mathsf{Pr}[BS_a(2i-1,2i) = \text{“00”} | BS_b(2i-1,2i) = \text{“01”}] =\\ &\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”} | BS_b(2i-1,2i) = \text{“01”}] = \frac{1}{2}\end{aligned}$$

Similarly, we can get that

$$\begin{aligned}&\mathsf{Pr}[BS_a(2i-1,2i) = \text{“00”} | BS_b(2i-1,2i) = \text{“10”}] =\\ &\mathsf{Pr}[BS_a(2i-1,2i) = \text{“11”} | BS_b(2i-1,2i) = \text{“10”}] = \frac{1}{2}\end{aligned}$$

From the above analysis, we can see that B gets no information about b_{1-s} from L_{1-s}. ■

Theorem 3. *When our OT$^2_1$ protocol is finished, A gets no information about s.*

Proof: (Sketch) First we observe that A does not know which bits in BS_b are different from the corresponding bits in BS_a. So it is easy to see that, for any $i \in I_0 \cup I_1$, whether $i \in I_0$ or $i \in I_1$ is independent from the distribution of $BS_b(2i-1,2i)$. So when the protocol is finished, A gets no information about s. ■

The above theorems demonstrate the security guarantees of our OT$^2_1$ protocol. Nevertheless, all these theorems are proved in the semi-honest model and under the assumption that the eavesdropper is passive. In practice, if the participants of OT$^2_1$ can deviate from the protocol, or if there is an active adversary launching a man-in-the-middle attack, then our OT$^2_1$ protocol needs to be modified and improved.

IV. Application I: Private Communications

In this section, we develop a method based on our OT$^2_1$ protocol that, assuming A and B both know a secret key K, allows A to send a confidential message to B. Our target here is similar to symmetric key encryption and decryption in traditional cryptography. More precisely, we have (at least) the following requirements for our private communications method:

- If both A and B use the same key, then B should get the message sent by A.
- If A and B use two different keys, then B does not get the message sent by A.
- Any eavesdropper gets no information about the message sent by A.

However, we stress that our method is only similar to, *not identical* to symmetric key encryption and decryption in traditional cryptography. The reason is that our communication model is completely different from that of traditional cryptography and so the security model is also different. For example, with our method, there is no ciphertext in the traditional sense. Hence, issues like chosen plaintext attack (which allows an adversary to see the ciphertexts for his chosen plaintexts) and chosen ciphertext attack are not considered for our method.

The idea underlying our method of private communications is very simple: Imagine that the keys used by A and B are of only one single bit, and the message to be sent is also a single bit. In this (unrealistic) situation, A can easily send the message to B by executing an OT$^2_1$ with B. In this OT$^2_1$, B's secret bit is his key, and A's secret bit b_K is set to her message, where K is A's key. It is easy to verify that our requirements listed above are all satisfied.

Of course, in a realistic scenario, the keys and the message are much longer. So we need to extend the above idea to multiple bits. Nevertheless, there is a pitfall that we must avoid: If we use a straightforward extension of the above idea (i.e., doing an OT$^2_1$ for each bit of the key, assuming the key and the message are of equal length.), and if A and B use two different keys, then B may end up getting part of the message sent by A, each bit of which corresponding to a bit position at which the two keys agree. To avoid this pitfall, we let A hide her message using a random mask, and then the mask is sent from A to B using a number of OT$^2_1$ sessions. Therefore, if A and B have different keys, the mask B receives will be different from what A sends at a number of bit positions (where the two keys differ). But when B attemps to recover the message using the wrong mask, the error in the recovered message will not remain at these bit positions; in stead, it will be spreaded over the entire message.

It is worth noting that not all properties of our OT$^2_1$ protocol are needed in the construction of our private communications method. In other words, our method of private communications can actually be simplified and optimized, from a practical point of view. We present it in the current form just to demonstrate the power of our OT$^2_1$ protocol.

Below is our method of private communications.

Let p be a prime of length k (where k is a parameter) that is well known, i.e., everybody knows p. Suppose that A and B both know a key K that is of length k. Recall that the objective is to send a confidential message M from A to B. Without loss of generality, suppose $M \in Z_p$. The method consists of three steps.

Step 1. A selects a mask D from $[0, 2^k - 1]$ uniformly at random. She then computes $C = (D \cdot M) \bmod p$, and sends C to B.

Step 2. Denote the jth bit of D by D_j, and the jth bit of K by K_j. For each $j \in [1, k]$, an OT_1^2 is executed between A and B, where A's two secret bits are $b_{K_j} = D_j$ and $b_{1-K_j} = 1 - D_j$, and B's secret bit is $s = K_j$.

Step 3. Once all the k OT_1^2 sessions are finished, B should have obtained all bits of D. Then B recovers M by computing $M = (C \cdot D^{-1}) \bmod p$.

The above private communications method is formally described in Algorithm 2.

Algorithm 2: Private Communications Method

```
Data: $p, k, K; M \in Z_p$.
Result: $B$ receives $M$.
$A$:
Select $D$ from $[0, 2^k - 1]$ uniformly at random.
$C \leftarrow (D \cdot M) \bmod p$.
Send $C$ to $B$.
foreach $j \in [1, k]$ do
    perform $\mathrm{OT}_1^2$ $[b_{K_j} = D_j, b_{1-K_j} = 1 - D_j; s = K_j]$ with $B$.
end
$B$:
$M = (C \cdot D^{-1}) \bmod p$
```

V. Application II: Privacy Preserving Password Verification

Besides private communications, our OT_1^2 protocol can also be applied to privacy preserving password verification. Today, password verification is still one of the major methods of user authentication. For example, in wireless LANs, many base stations authenticate users using their passwords at the beginning of sessions. However, it is clear that, when users send their passwords through wireless links, there is a risk that the passwords may be overheard by an adversary. Furthermore, an adversary may impersonate a base station or a password protected server to ask users for their passwords. Hence, it is important to consider the privacy protection of passwords when we use passwords for authentication.

In this section, we study privacy preserving password verification, which allows one wireless device to verify the password from another wireless device without the risk of revealing the password. More precisely, we have the following requirements when B verifies the password of A.

- If A's password matches the corresponding password in B's record, then B should accept.
- If A's password does not match the corresponding password in B's record, then B should reject.
- In any case, A learns nothing about the password in B's record except whether it matches A's password or not.
- In any case, B learns nothing about A's password except whether it matches the corresponding password in B's record or not.
- An eavesdropper should not learn anything about either A's password or the password in B's record.

In the above, the fourth requirement guarantees that, even if B is corrupted by an adversary, B will not be able to learn A's password as long as B has not already known it. (Of course, a corrupted device B might launch a probe attack, by repeatedly requesting A to do password authentication. Nevertheless, this is easy to prevent if A is required to stop trying after a few number of times.) So the fourth and fifth requirements together give a strong privacy protection for A's password. Similarly, the third and fifth requirements together give a strong privacy protection for the password in B's record.

To achieve the above objective, our main idea is to let A generate l pairs of random numbers and then execute OT_1^2 with B. After these OT_1^2 , B receives one out of each pair of random numbers. So in total, B receives a sequence of l random numbers. Clearly, there are altogether 2^l such sequences, from which B choose to receive one. Among these 2^l sequences, only one sequence satisfies a special property: The product of all random numbers in this sequence is congruent to 1 (with respect to a prime modulus p). B will receive this special sequence through these OT_1^2 if and only if A's password matches the password in B's record. Therefore, in order to verify A's password, B only need to verify that the received sequence satisfies the special property described above.

Below are the details of our privacy preserving method for password verification.

Just like in Application I, let p be a well-known prime of length k, where k is a parameter. Without loss of generality, suppose that each password is of length l, where l is another parameter. Let Pass be A's password.

Step 1. A sends her user identity to B. Using this identity, B finds the corresponding password in B's record. Suppose that what B finds is Pass'.

Step 2. Denote by $\mathsf{Pass_i}$ (resp., $\mathsf{Pass_i'}$) the ith bit of Pass (resp., Pass'). For each $i \in [1, l-1]$, A picks two random numbers $\beta_{0,i}, \beta_{1,i} \in Z_p$ independently and uniformly. Finally, A computes

$$\beta_{\mathsf{Pass}_\ell,\ell} = (\prod_{i=1}^{l-1} \beta_{\mathsf{Pass_i},\mathsf{i}})^{-1} \pmod{p},$$

and picks $\beta_{1-\mathsf{Pass}_\ell,\ell} \in Z_p$ uniformly and independently.

Step 3. Denote by $\beta_{0,i,j}$ (resp., $\beta_{1,i,j}$) the jth bit of $\beta_{0,i}$ (resp., $\beta_{1,i}$). For each $i \in [1, l]$ and each $j \in [1, k]$, A and B execute an OT_1^2 , where A's two secret bits are $\beta_{0,i,j}$ and $\beta_{1,i,j}$, and B's secret bit is $\mathsf{Pass_i'}$; let $\beta'_{i,j}$ be what B receives in the OT_1^2 .

Step 4. For each i, B puts together the k bits $\beta'_{i,1}$, $\beta'_{i,2}$, $\ldots$, $\beta'_{i,k}$ to get an integer β'_i. Then, B verifies that

$$\prod_{i=1}^{l} \beta'_i \equiv 1 \pmod{p}.$$

A formal description of the above privacy preserving method for password verification is given in Algorithm 3.

Algorithm 3: Privacy Preserving Password Verification

```
Data: Pass, Pass′, $p$, $k$, $l$.
Result: If Pass=Pass′, then $B$ accepts $A$'s authentication
        request; otherwise $B$ rejects $A$'s authentication request.

$A$:
foreach $i \in [1, l-1]$ do
    pick two random numbers $\beta_{0,i}, \beta_{1,i} \in Z_p$ independently
    and uniformly.
end
$\beta_{\mathsf{Pass}_\ell,\ell} \leftarrow (\prod_{i=1}^{l-1} \beta_{\mathsf{Pass_i},\mathsf{i}})^{-1} \pmod{p}$.
Pick $\beta_{1-\mathsf{Pass}_\ell,\ell} \in Z_p$ uniformly and independently.
foreach $i \in [1, l]$ do
    foreach $j \in [1, k]$ do
        perform $\mathrm{OT}_1^2$ $[\beta_{0,i,j}, \beta_{1,i,j}; \mathsf{Pass}'_\mathsf{i}]$ with $B$. (Denote
        the bit $B$ receives by $\beta'_{i,j}$.)
    end
end

$B$:
foreach $i \in [1, l]$ do
    Combine $\beta'_{i,1}$, $\beta'_{i,2}$, $\ldots$, $\beta'_{i,k}$ to get $\beta'_i$
end
if $\prod_{i=1}^{l} \beta'_i \equiv 1 \pmod{p}$ then
    accept $A$'s authentication request.
end
else
    reject $A$'s authentication request.
end
```

VI. Implementation and Evaluations

We completely implement our OT_1^2 protocol on two laptops, one with Intel Core2 CPU of 2.33GHz and 2.0 GB memory, the other with Intel Pentium M CPU of 2.13GHz and 1001.5 MB memory. Both laptops run the Ubuntu Linux 9.10 operating system and use Netgear WAG511 802.11abg wireless network cards. Both cards use ath5k [1] as drivers and are configured to operate in the 802.11a frequency band (specifically, the 5.745GHz frequency band). The transmission power is set to be 30dBm for both cards.

In order that the two laptops can communicate directly without any intermediate relays, we configure one laptop in the access point (AP) mode, and configure the other laptop in the station mode. ICMP echo request packets are sent from the station to the AP at a constant rate. Once the AP receives the packet, it sends an ICMP echo reply packet back to the station.

We create one monitor interface on each of the two laptops, so that we can use tcpdump [3] to capture the packets. By customizing the tcpdump filters, we capture only ICMP echo request packets on the AP side and only ICMP echo reply packets on the station side. The received signal strength (RSS) in the radiotap header [2] is extracted from each captured packet. Because the transmission power levels for both sides are identical, the extracted RSS is a coarse measurement of the amplitude of wireless channel. (Ideally, rather than using RSS, our experiments should use raw physical layer complex channel impulse responses. However, in order to perform our experiments on *off-the-shelf* 802.11 network cards, we choose to use RSS, just like in [32], [26].) Each of the RSS measurement is quantized into one bit.

As pointed out in [32], [26], large-scale shadow fading can lead to long sequences of zeros and ones in the extracted bit strings. Mathur et al. [32] eliminate this effect by subtracting a moving average signal strength from the extracted RSS values, while Jana et al. [26] solve the problem using an adaptive quantization method. Similar to [26], we also use the adaptive quantization method in our experiments.

Our experiments are carried out in two settings. In the first setting, the two laptops are stationary. In the second setting, the station moves. In each setting, we measure the RSS profiles at both sides and also the minimum number of channel probings needed for an OT_1^2 . The results are presented in Sections VI-A and VI-B, respectively.

Besides the above experiments on RSS and the minimum number of channel probings, we have also experimentally studied the efficiency of our OT_1^2 prototol. The results are given in Section VI-C.

In addition, we have also implemented the private communications method based on our OT_1^2 protocol. The evaluations of this application are presented in Section VI-D.

A. OT_1^2 between Stationary Devices

In the first setting, we place the two laptops at fixed locations. Specifically, we place them on two tables in a library, and the distance between them is 15 meters. A number of people are walking in the library at speeds of 0.5–1m/s, which causes variations in the wireless channel between the AP and the station. This environment is illustrated in Fig. 1.

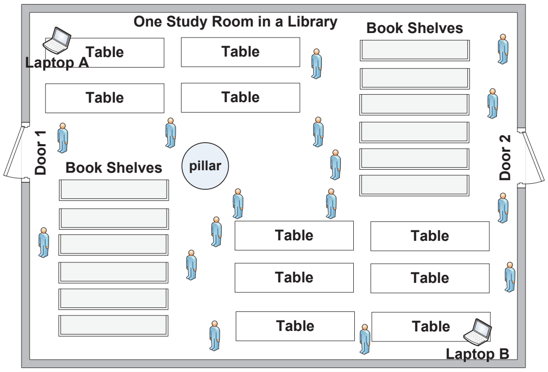

Fig. 1. The environment in the first setting.

In this setting, we first do an experiment to measure the RSS, which lasts for 300 seconds. During these 300 seconds, each laptop sends one probe signal every 100 milliseconds. From the captured packets, the RSS values are extracted and quantized into bit strings. Note that at both laptops we have implemented mechanisms to deal with packet losses and retransmissions, so that lost packets are removed from considerations and retransmitted packets are not repeatedly counted.

The extracted RSS sequences in the above experiment are shown in Fig. 2, from which we can see that the signal strengths at the station are always greater than the signal strengths at the AP. The reason is that the two wireless network cards are a little different in terms of signal sensitivities. The card of the AP has a noise level of -100dB, while the card of the station has a noise level of -98dB. However, the absolute values of signal strengths do not have any influence on our OT_1^2 protocol because the quantization thresholds are computed based on the local mean and the standard deviation of the measured RSS sequences.

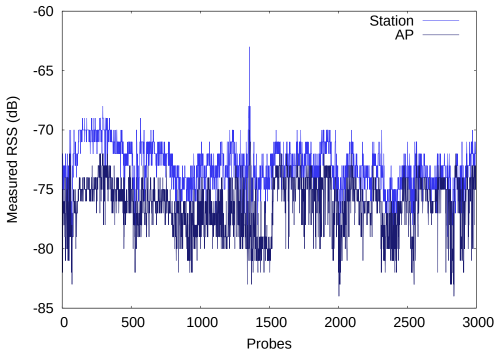

Fig. 2. Measured RSS profiles—the stationary setting.

Next, we do a number of experiments to measure the minimum number of channel probings required to achieve a certain error probability. (Here by error probability we mean the probability that the received bit in an OT_1^2 is not equal to b_s.) We repeat our experiment for different error probabilities between 0.01 and 0.0001, and for different combinations of quantization parameters m and α. Fig. 3 shows our results. We can see that, to achieve an error probability of 10^{-3}, we only need about 150 channel probings when $m = 50$ and $\alpha = 0.25$.

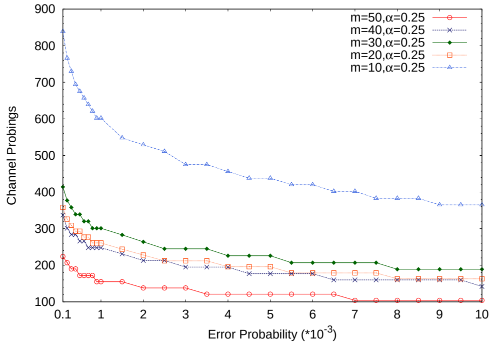

Fig. 3. The minimum channel probings to achieve required error probabilities—the stationary setting.

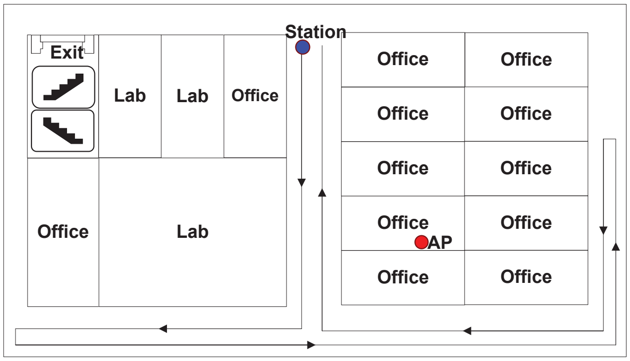

Fig. 4. The environment in the second setting

B. OT_1^2 with Moving Station

In the second setting, we place the AP on a table, and let the station move at a speed of 1 m/s. The environment of this setting is shown in Fig. 4. Because the network cards are set to send and receive data in the 5.745GHz frequency band, we can calculate the approximate channel coherence time according to the following equation, in which c is the speed of light and f is the central transmission frequency.

$$T_C \approx \frac{\lambda}{v} = \frac{c}{f} = \frac{3 \cdot 10^8 \ m/s}{5.745 \cdot 10^9 \ Hz} \approx 52.219 \ ms.$$

In this setting, we first do an experiment to measure the RSS, which lasts for about 160 seconds. During these 160 seconds, each laptop sends one probe signal every 53 milliseconds. From the captured packets, the RSS values are extracted and quantized into bit strings. The results are given in Fig. 5. We can see that due to the relative speed of 1 m/s, there are more major fluctuations of signal strengths than in the first setting.

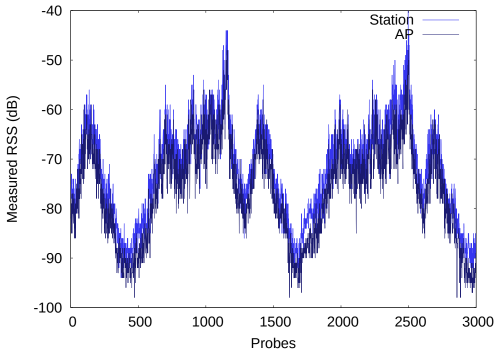

Fig. 5. Measured RSS profiles—the mobile setting.

Next, just like in the first setting, we do a number of experiments to measure the minimum number of channel probings required to achieve a certain error probability. We repeat our experiment for different error probabilities between 0.01 and 0.0001, and for different combinations of quantization parameters m and α. Fig. 6 gives our results. We can see that, to achieve an error probability of 10^{-3}, we only need about 100 channel probings when $m = 50$ and $\alpha = 0.2$.

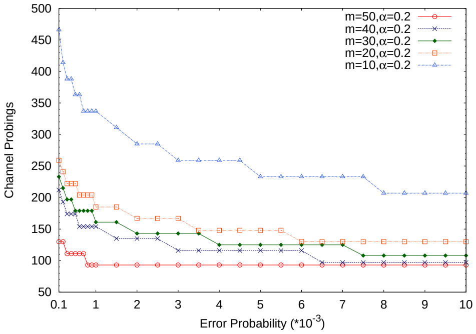

Fig. 6. The minimum channel probings to achieve required error probabilities – the mobile setting.

C. *Efficiency of* OT_1^2

To test the efficiency of our OT_1^2 protocol, we run the protocol for 1000 times and measure the average running time. In this experiment, we choose m=50 and α=0.2, and each execution of the protocol includes 100 channel probings. Since the protocol efficiency is directly affected by the coherence time of the wireless channel, we make measurements for different values of channel coherence time. The result is shown in Fig. 7. Because the channel coherence time is affected by the relative speed, for ease of understanding, we also include the corresponding values of relative speed in the figure.

From Fig. 7 we can see that our OT_1^2 protocol can be completed within several seconds if one participant moves relatively to the other at a normal walking speed. When the relative speed increases, the protocol execution time decreases very quickly. For example, at a typical city driving speed of 20 $\sim$ 40 mph (8.9 $\sim$ 17.9 m/s), the OT_1^2 protocol can be finished in less than 1 second.

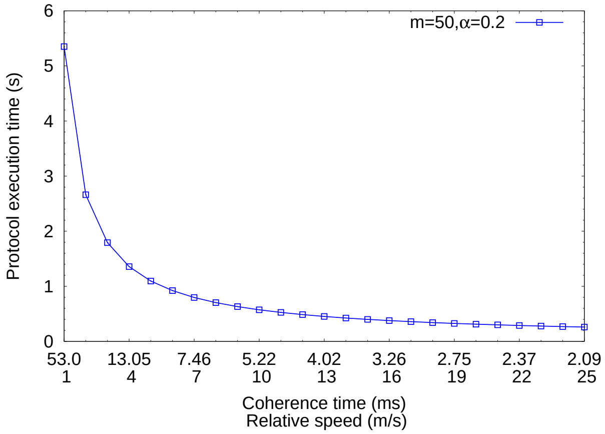

Fig. 7. OT_1^2 protocol execution time for different values of coherence time and relative speed.

D. *Evaluations of the Private Communications Method*

We also implement the private communications method described in Section IV and evaluate it experimentally.

Specifically, we choose $k = 128$ and privately transmit a 128-bit message from the station to the AP. We consider the transmission successful if the recovered message at the AP is the same with the message sent by the station. We try transmitting 50 messages in our experiment and all of them are successful.

The efficiency of our private communications method is illustrated in Fig. 8. Again, we make our measurements of the execution time for different values of the coherence time. From the results we can see that, if the coherence time is 5.2 ms (which can be achieved when the relative speed is 10 m/s), the total execution time is less than 80 seconds. We admit that this may not be as fast as private communications based on traditional cryptography. However, if the transmitted message is security critical, then we may want to consider sacrificing some efficiency in order to prevent possible privacy violation in the future (when the used cryptosystem is broken).

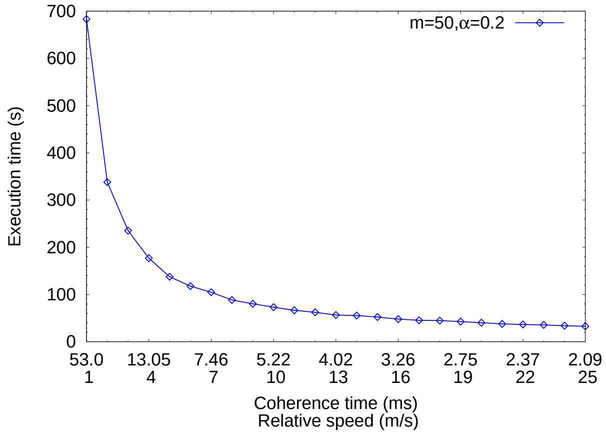

Fig. 8. Execution time of private communications method for different values of coherence time and relative speed.

VII. Related Work

As we have mentioned, our work is motivated by the previous works on key agreement using wireless channel characteristics. In [33], [4], it is shown that secure key agreement can be achieved using the correlated information between two wireless devices as long as they share an authenticated channel beforehand. In [22], Hershey et al. propose a key agreement protocol that extracts secret bits from phase differences of continuous waves. After that, many other methods [21], [40], [5], [30], [45], [6], [32], [39], [26], [34] are proposed to enhance the security and/or improve the performance. In particular, Li et al. [30] propose a set of wireless security mechanisms, including wireless channel-based authentication, key extraction and key dissemination. In [6], Azimi et al. propose to achieve key agreement by quantizing the deep fading in mobile radio channels. The technique of information reconciliation [9] is used to make the extracted keys consistent, while privacy amplification [7], [24], [10] is used to remove side information leaked during information reconciliation.

Recently, Mathur et al. [32] propose a very practical method for secret key extraction from an unauthenticated wireless channel. They design a level crossing algorithm for achieving key agreement between the protocol participants. Their method is resistant to spoofing attack. To improve the secret bit rate efficiently, Jana et al. [26] design an adaptive and multi-bit quantization method for secret bit extraction. They do extensive experiments under a diversity of environments and make comparisons among them. In [34], a high rate uncorrelated bit extraction scheme is proposed, which further improves the efficiency by using fractional interpolation, de-correlation transformation and multi-bit adaptive quantization. Another recent work by Ye et al. [44] presents improvements in both efficiency and generality of channel state distributions.

While the aforementioned works are on key agreement, our work is on oblivious transfer (OT), or more precisely, OT_1^2 . OT is a fundamental cryptographic tool that has been used in constructions of many complex cryptographic protocols. It is first proposed by Rabin [36]. Even, Goldreich and Lempel [16] propose OT_1^2 , an important variant of OT. Crepeau [11] shows that OT_1^2 is equivalent to the original version of OT proposed by Rabin. The importance of OT is reflected by its completeness [28], [19], [13], [25]. In his seminal work, Kilian [28] shows that any general two-party cryptographic protocol can be built using OT. In [19], [13], [25], this result is extended to multiparty protocols.

In a theoretical work [14], Crepeau and Kilian propose an OT_1^2 protocol based on noisy channels. Crepeau also proposes another OT_1^2 protocol in a follow-up work [12] to increase the efficiency. The noisy channels they consider are simple discrete memoryless channels. In contrast, our OT_1^2 protocol is based on wireless channels, which are much more realistic and complicated, having severe fluctuations with varying time and locations. Furthermore, in addition to theoretical analysis, we have fully implemented our OT_1^2 protocols with off-the-shelf 802.11 network cards and carried out extensive experiments.

The first application of our OT_1^2 protocol is private communications. In fact, there have been a number of works on private communications based on wireless channel characteristics, e.g., [29], [27], [20], [8], [31], [35], [15], among others. We stress that our private communications method is to illustrate the application of our OT_1^2 protocol. We choose this application because it is simple and easy to understand, *not* because our private communications method is more efficient than the existing works on private communications.

VIII. Conclusion

In this paper, we propose an OT_1^2 protocol in the setting of a wireless network and give two applications of this protocol to illustrate its potential broad applications. The main advantage of our OT_1^2 protocol is that it does not rely on any computational assumption. For security critical applications in wireless networks, such an advantage is of great importance, because as we have seen in the history, cryptographic tools based on computational assumptions may be broken after being used for years.

Although at this moment, our OT_1^2 protocol is still not as fast as the traditional OT_1^2 protocols based on computational assumptions, it has shown the feasibility of basing wireless security on physical channel characteristics, rather than on computational assumptions. Hence, our work can be considered a crucial step towards buiding wireless security systems that do not rely on computational assumptions.

In terms of security, our OT_1^2 protocol and its applications are secure in the semi-honest model, and under the assumption that there is only a passive eavesdropper besides the protocol participants. We leave the consideration of fully malicious model and/or active man-in-the-middle attack to future work.

References

[1] Ath5k Linux Wireless Network Card Driver. http://linuxwireless.org/en/users/Drivers/ath5k.

[2] Radiotap Header. http://www.radiotap.org/.

[3] Tcpdump. http://www.tcpdump.org/.

[4] R. Ahlswede and I. Csiszar. Common randomness in information theory and cryptography. i. secret sharing. *Information Theory, IEEE Transactions on*, 39(4):1121 –1132, Jul 1993.

[5] T. Aono, K. Higuchi, T. Ohira, B. Komiyama, and H. Sasaoka. Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. *Antennas and Propagation, IEEE Transactions on*, 53(11):3776–3784, Nov. 2005.

[6] B. Azimi-Sadjadi, A. Kiayias, A. Mercado, and B. Yener. Robust key generation from signal envelopes in wireless networks. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 401–410, New York, NY, USA, 2007. ACM.

[7] C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, 1988.

[8] M. Bloch and A. Thangaraj. Confidential messages to a cooperative relay. In *Information Theory Workshop, 2008. ITW '08. IEEE*, pages 154 –158, May 2008.

[9] G. Brassard and L. Salvail. Secret-key reconciliation by public discussion. In *EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology*, pages 410–423, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[10] C. Cachin and U. M. Maurer. Linking information reconciliation and privacy amplification. *Journal of Cryptology*, 10(2):97–110, 3 1997.

[11] C. Crépeau. Equivalence between two flavours of oblivious transfer. *In Advances in Cryptology: CRYPTO '87*, 293.

[12] C. Crépeau. Efficient cryptographic protocols based on noisy channels. In *EUROCRYPT'1997*, pages 306–317, 1997.

[13] C. Crépeau, J. v. d. Graaf, and A. Tapp. Committed oblivious transfer and private multi-party computation. In *CRYPTO '95: Proceedings of the 15th Annual International Cryptology Conference on Advances in Cryptology*, pages 110–123, London, UK, 1995. Springer-Verlag.

[14] C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions. In *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 42–52, Oct 1988.

[15] L. Dong, Z. Han, A. P. Petropulu, and H. V. Poor. Improving wireless physical layer security via cooperating relays. *Signal Processing, IEEE Transactions on*, 58(3):1875 –1888, March 2010.

[16] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.

[17] FIPS Publication 197. *Advanced Encryption Standard*. NIST, November 2001.

[18] O. Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2004.

[19] S. Goldwasser and L. A. Levin. Fair computation of general functions in presence of immoral majority. In *CRYPTO '90*, pages 77–93, 1991.

[20] P. Gopala, L. Lai, and H. El Gamal. On the secrecy capacity of fading channels. *Information Theory, IEEE Transactions on*, 54(10):4687 – 4698, Oct. 2008.

[21] A. A. Hassan, W. E. Stark, J. E. Hershey, and S. Chennakeshu. Cryptographic key agreement for mobile radio. *Digital Signal Processing*, 6(4):207 – 212, 1996.

[22] J. Hershey, A. Hassan, and R. Yarlagadda. Unconventional cryptographic keying variable management. *Communications, IEEE Transactions on*, 43(1):3–6, Jan 1995.

[23] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[24] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 12–24, New York, NY, USA, 1989. ACM.

[25] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer — efficiently. In *CRYPTO 2008: Proceedings of the 28th Annual conference on Cryptology*, pages 572–591, Berlin, Heidelberg, 2008. Springer-Verlag.

[26] 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 '09*, pages 321–332, 2009.

[27] A. Khisti, A. Tchamkerten, and G. Wornell. Secure broadcasting over fading channels. *Information Theory, IEEE Transactions on*, 54(6):2453 –2469, June 2008.

[28] J. Kilian. Founding crytpography on oblivious transfer. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 20–31, New York, NY, USA, 1988. ACM.

[29] L. Lai and H. El Gamal. The relay-eavesdropper channel: Cooperation for secrecy. *Information Theory, IEEE Transactions on*, 54(9):4005 – 4019, Sept. 2008.

[30] Z. Li, W. Xu, R. Miller, and W. Trappe. Securing wireless systems via lower layer enforcements. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 33–42, New York, NY, USA, 2006. ACM.

[31] Y. Liang, H. Poor, and S. Shamai. Secure communication over fading channels. *Information Theory, IEEE Transactions on*, 54(6):2470–2492, June 2008.

[32] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 128–139, New York, NY, USA, 2008. ACM.

[33] U. Maurer. Secret key agreement by public discussion from common information. *Information Theory, IEEE Transactions on*, 39(3):733 –742, May 1993.

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

[35] E. Perron, S. Diggavi, and E. Telatar. On cooperative wireless network secrecy. In *INFOCOM 2009, IEEE*, pages 1935 –1943, April 2009.

[36] M. O. Rabin. How To Exchange Secrets with Oblivious Transfer. *Technical Report TR-81, Aiken Computation Lab, Harvard University*, 1981.

[37] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[38] S. Ross. *A First Course in Probability*. Prentice Hall, 2002.

[39] A. Sayeed and A. Perrig. Secure wireless communications: Secret keys through multipath. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3013–3016, 2008.

[40] M. Tope and J. McEachen. Unconditionally secure communications over fading channels. In *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, volume 1, pages 54–58 vol.1, 2001.

[41] J. Tugnait, L. Tong, and Z. Ding. Single-user channel estimation and equalization. *Signal Processing Magazine, IEEE*, 17(3):16–28, May 2000.

[42] X. Wang and H. Yu. How to Break MD5 and Other Hash Functions. In *EUROCRYPT 2005*, pages 19–35. Springer Berlin / Heidelberg, 2005.

[43] X. Wang, H. Yu, and Y. L. Yin. Efficient Collision Search Attacks on SHA-0. In *CRYPTO 2005*, pages 1–16. Springer Berlin / Heidelberg, 2005.

[44] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. Mandayam. Information-theoretically secret key generation for fading wireless channels. *Information Forensics and Security, IEEE Transactions on*, 2010. To appear.

[45] C. Ye, A. Reznik, and Y. Shah. Extracting secrecy from jointly gaussian random variables. In *Information Theory, 2006 IEEE International Symposium on*, pages 2593–2597, July 2006.