Exchanging Pairwise Secrets Efficiently

Iris Safaka, Christina Fragouli, Katerina Argyraki
EPFL, Switzerland

Suhas Diggavi
UCLA

***Abstract*—We consider the problem where a group of wireless nodes, connected to the same broadcast domain, want to create pairwise secrets, in the presence of an adversary Eve, who tries to listen in and steal these secrets. Existing solutions assume that Eve cannot perform certain computations (e.g., large-integer factorization) in useful time. We ask the question: can we solve this problem without assuming anything about Eve's computational capabilities?**

We propose a simple secret-agreement protocol, where the wireless nodes keep exchanging bits until they have agreed on pairwise secrets that Eve cannot reconstruct with very high probability. Our protocol relies on Eve's limited network presence (the fact that she cannot be located at an arbitrary number of points in the network at the same time), but assumes nothing about her computational capabilities. We formally show that, under standard theoretical assumptions, our protocol is information-theoretically secure (it leaks zero information to Eve about the secrets). Using a small wireless testbed of smartphones, we provide experimental evidence that it is feasible for 5 nodes to create thousands of secret bits per second, with their secrecy being independent from the adversary's capabilities.

I. Introduction

We consider the problem where a group of n nodes, connected to the same wireless broadcast domain, want to create pairwise secrets, such that an adversary Eve, who is eavesdropping on their domain, obtains very little information on the secrets. Today, we can solve this problem only by relying on Eve's computational limitations: we can use an asymmetric key-agreement protocol, like RSA [1], which fundamentally assume that Eve cannot perform certain computations, such as large-integer factorization.

Wireless networks offer the opportunity for a different, complementary kind of security, which relies not on the adversary's computational limitations, but on her limited network presence. Suppose Alice, Bob, and Eve are wireless nodes connected to the same broadcast domain. When Alice transmits, both Bob and Eve will overhear a part of her transmission, however, as long as there is sufficient noise, it is unlikely that they will overhear exactly the same information. It has been long known in the information theory community that, if Bob and Eve do not overhear exactly the same information from Alice's transmission, it is theoretically possible for Alice and Bob to create a shared secret Eve knows nothing about, even if Eve has arbitrary computational capabilities [2], [3]. The question is, can we turn this theoretical feasibility result into a concrete secret-agreement protocol? how well would such a protocol work on an actual wireless network?

First, we present a secret-agreement protocol, which enables n nodes to create pairwise secrets that Eve knows very little about. Our protocol has polynomial complexity and is implementable in simple wireless devices. Also, it creates all the pairwise secrets *simultaneously* by harnessing the broadcast nature of wireless networks. The latter property enables secret agreement to achieve a significant secret-generation rate even with large numbers of nodes (to the best of our knowledge, we are the first to make this observation). Under standard information-theory assumptions (independent erasure channels between the nodes and known erasure probabilities), we formally show that: (1) Our protocol is information-theoretically secure, i.e., it leaks no information to Eve about the secrets. (2) It achieves a secret-generation rate that is optimal for $n = 2$ nodes and scales well with the number of nodes n.

Second, we provide experimental evidence that it is feasible to use our protocol in practice to create thousands of secret bits per second. In particular, we use a small wireless testbed that consists of 5 nodes and an adversary, located in adjacent offices. In our testbed, the nodes create pairwise secrets at rates of thousands of bits per second, independently from the adversary's computational capabilities.

Unlike current security systems, our protocol creates secrets that do not depend on any information permanently stored at the nodes—only on traffic exchanged among them. Hence, the nodes could use our protocol to periodically create new secrets, then use these secrets to constantly refresh their encryption and authentication keys. The nodes do not need to maintain any public/private RSA key pair or WPA master key that, if stolen or accidentally revealed to an adversary, would enable her to reconstruct the secrets (hence compromise the nodes' communications).

Most importantly, we shift the adversary's challenge from computation to network presence: for current systems, a dangerous adversary is one with high computational power (e.g., one with access to quantum computers); for our protocol, a dangerous adversary is one who is physically present in many locations in the network at the same time.

We do not advocate to replace the existing crypto-systems that rely on the adversary's computational limitations. However, we believe that exploring alternative approaches (which rely on different kinds of adversary limitations) will become of increasing interest in the near future, as governments and corporations acquire massive computational capabilities. Interest in alternatives is already present in the industry community, where several companies are developing quantum key distribution (QKD) systems [4]. A typical application envisioned for these systems is the periodic generation of one-time pads at a high enough rate to enable information-theoretically secure transmission of real-time video, e.g., for military operations [5]. Unfortunately, QKD systems are ex-

I. Safaka is supported by ERC Starting Grant ERC-2009-StG-240317.
S. Diggavi is partly supported by AFOSR MURI, award FA9550-09-064.

pensive (due to the need for sophisticated equipment such as photon detectors) and therefore accessible only to the wealthiest governments and corporations. This motivated us to explore the feasibility of a secret-agreement protocol that neither relies on computational limitations nor requires expensive equipment. We focus on wireless devices, as they form the majority of network-connected devices. We consider groups of such devices (as opposed to isolated pairs), as wireless group cooperation schemes are becoming increasingly popular to support social applications [6]. As we will show, it turns out that by simultaneously creating multiple secrets we achieve a multi-fold increase in efficiency.

After stating our problem (§II), we first describe our basic secret-agreement protocol, which enables n nodes to create pairwise secrets under standard theoretical assumptions (§III), and we state its properties (§IV). Next, we adapt our protocol to the scenario where the theoretical assumptions do not hold (§V), and provide experimental evidence of its capabilities (§VI). Finally, we summarize related work (§VII), and we conclude with a discussion of the remaining challenges (§VIII).

II. Setup

A. Problem Statement

We consider n nodes, $T_1, \ldots, T_n$, connected to the same wireless broadcast domain. We will refer to these nodes as *terminals*. Sometimes we will refer to terminals T_1, T_2, T_3, and T_4 respectively as Alice, Bob, Calvin, and David.

We consider an adversary, Eve, connected to the same broadcast domain as the terminals. Our design assumes that Eve may possess up to some number of receiving antennas m. But we should state upfront that the experimental results presented in Section V assume that Eve is an HTC smartphone with one omnidirectional antenna.

The terminals communicate with each other in two ways:

- When we say that terminal T_i *transmits* a packet, we mean that it broadcasts the packet once.
- When we say that terminal T_i *reliably broadcasts* a packet, we mean that it ensures that all other terminals $T_{j\neq i}$ receive it, e.g., through ACKs and retransmissions.

To be conservative, we assume that Eve receives all reliably broadcast packets.

Our goal is to design a protocol that enables each terminal pair, T_i and T_j, to create a secret $\mathcal{S}_{ij}$, such that any other terminal $T_{l\neq i,j}$ or Eve obtain very little information on $\mathcal{S}_{ij}$.

We assume that each terminal T_i is "honest but curious" toward the other terminals. I.e., T_i runs the protocol honestly but may try to eavesdrop on other terminals' communications.

We assume that Eve may be a passive adversary (never makes any transmissions) or an active one (may try to impersonate a terminal). If Eve is passive, the terminals do not need to share any information before they run our protocol. If Eve is active, the terminals need to initially share some bootstrap information, in order to authenticate each other when they first communicate with each other (until they create their first pairwise secrets using our protocol). The need for this bootstrap information is fundamentally unavoidable: without it, there is no way for Alice to know that she is talking to Bob until they have established their first secret. However, any pairwise secrets created through the protocol should not depend on the bootstrap information.

Symbol	Meaning
n	Number of terminals
T_i	Terminal i
$\mathcal{S}_{ij}$	Secret between terminals T_i and T_j
δ_{ij}	Erasure probability of T_i – T_j channel
δ_{iE}	Erasure probability of T_i – Eve channel
N	Number of x-packets transmitted by each terminal (initial phase, step 1)
M_{ij}	Number of shared y-packets constructed by T_i and T_j (privacy amplification phase, steps 1 – 3)

TABLE I
COMMONLY USED SYMBOLS

This setup corresponds to the scenario where a group of n political dissidents rendezvous in a public place (potentially under visual surveillance) and use their cell phones in ad-hoc mode to secretly communicate; or the scenario where a group of n friends connect to the same social network and use their cell phones in ad-hoc mode to exchange private content. It should be infeasible for an eavesdropper who listens in on the same broadcast domain to record what she overhears, process the recording, and reconstruct their communications. Moreover, it should be infeasible for an eavesdropper to record what she overhears, extract from the dissidents/friends a set of passwords or keys, combine them with the recording, and reconstruct their communications. The dissidents/friends can periodically use our protocol to create pairwise secrets and use these secrets to continuously refresh the keys with which they encrypt/authenticate their communications.

B. Theoretical Network Conditions

We define them as follows:

1) When terminal T_i transmits a packet, terminal T_j (Eve):
   - misses the entire packet, with probability δ_{ij} (δ_{iE})
   - receives the entire packet correctly, otherwise.

   δ_{ij} (δ_{iE}) is the *erasure probability* of the T_i – T_j (T_i – Eve) channel.
2) The T_i – T_j channel is independent from any T_i – $T_{l\neq j}$ channel[1] and the T_i – Eve channel, for all i, j, l.
3) The erasure probability δ_{iE} of the T_i – Eve channel is known, for all i.

C. Quality Metrics

Efficiency captures the cost of the protocol, i.e., the amount of traffic it produces in order to generate pairwise secrets of a given size. The efficiency achieved by two terminals T_i and T_j that create a secret $\mathcal{S}_{ij}$ is defined as:

$$E_{ij} = \frac{|\mathcal{S}_{ij}|}{transmitted\ \ bits}.$$

[1]Assuming independent channels is not necessary for any of our results, but simplifies our proofs.

The denominator is the total number of bits transmitted by the protocol until $\mathcal{S}_{ij}$ is created.

The *secrecy rate* achieved by T_i and T_j is the number of secret bits per second that they create between them. It depends on the transmission rates of the terminals and their efficiency.

Reliability captures the quality of the created secrets, i.e., the extent to which they are unknown to Eve. The reliability of a secret $\mathcal{S}$ is defined as:

$$R = \frac{H(\mathcal{S}|\mathcal{X}_E)}{H(\mathcal{S})},$$

where $\mathcal{X}_E$ is the information obtained by Eve via eavesdropping on the terminals' communications, $H(\mathcal{S})$ is Eve's entropy (her uncertainty about $\mathcal{S}$ before she eavesdrops), and $H(\mathcal{S}|\mathcal{X}_E)$ is Eve's conditional entropy (her uncertainty about $\mathcal{S}$ after she eavesdrops). $R = 1$ means that Eve learns nothing about $\mathcal{S}$ by eavesdropping (which would mean that the protocol is information-theoretically secure).

III. Basic Protocol

In this section, we describe the core of our secret-agreement protocol, which enables terminals T_i and T_j to create a secret $\mathcal{S}_{ij}$. Assuming the theoretical network conditions, $\mathcal{S}_{ij}$ is perfectly secret from any terminal $T_{k\neq i,j}$ and an adversary Eve (we show this in Section IV).

A. Basic Idea

Suppose Alice and Bob exchange three packets, x_1, x_2 and x_3. Suppose Eve misses (knows nothing about the contents of) two of the packets shared by Alice and Bob, x_1 and x_2. If an oracle told Alice and Bob that Eve misses two of their shared packets (but not which two), they could create a perfect shared secret (one that Eve knows nothing about), by using two linear combinations of their shared packets, e.g., $\langle x_1 + x_2, x_2 + x_3 \rangle$[2] (where $+$ denotes addition over a finite field, e.g., bitwise XOR over the binary field).

Building on this idea, our protocol consists of two phases: In the *initial* phase, the terminals exchange traffic to ensure that each terminal pair shares some number of packets (as Alice and Bob share x_1, x_2, and x_3 in the above example). This happens over n rounds, with a different terminal transmitting in each round. In the *privacy amplification* phase, each terminal pair creates a secret out of the information they shared in the initial phase. For this, they "compress" their shared information enough to ensure that any other terminal or Eve know nothing about the secret (as Alice and Bob "compress" x_1, x_2, and x_3 into $x_1 + x_2$, $x_2 + x_3$ in the above example). To do this compression correctly, the terminals need to know how much of their traffic exchange was overheard by Eve (but not which particular bits).

A naive approach would be to have each terminal pair create their secret separately, which would not scale well with the number of terminals. Instead, our protocol creates the pairwise secrets simultaneously, by harnessing the broadcast nature of wireless networks.

[2]This secret is perfect, because Eve's probability of guessing its value is equal to the probability of guessing the values of the two packets she misses.

B. Algorithm

Each terminal T_i maintains $n-1$ queues $\mathcal{Q}_{ij}$, $j \neq i$. In the beginning, these are empty.

Initial Phase

In round $k = 1 \ldots n$:

1) Terminal T_k transmits N random packets (we will call them x-*packets*).
2) Each terminal $T_{i\neq k}$ reliably broadcasts the identities of the x-packets it received.
3) Each terminal T_i adds to queue $\mathcal{Q}_{ij}$ the identities and contents of the x-packets it shares with terminal $T_{j\neq i}$.

At this point, $\mathcal{Q}_{ij}$ contains all the packets shared by terminals T_i and T_j.

Privacy Amplification Phase

For $i = 1 \ldots n-1$:

1) Terminal T_i constructs M_{ij} linear combinations of the packets in the queue $\mathcal{Q}_{ij}$, for all $j > i$ (we will call them y-*packets*).
   It determines the number of y-packets M_{ij} and constructs the y-packets as described in Section III-E.
2) Terminal T_i reliably broadcasts the coefficients it used to construct the y-packets.
3) Each terminal $T_{j>i}$ uses the broadcasted coefficients and the contents of its queue $\mathcal{Q}_{ji}$ to reconstruct the M_{ij} y-packets.

At this point, terminals T_i and $T_{j>i}$ share M_{ij} y-packets. Their secret $\mathcal{S}_{ij}$ is the concatenation of these y-packets.

C. An Example Agreement

Suppose we have $n = 3$ terminals, Alice, Bob, and Calvin, and a passive adversary, Eve. All the channels between terminals or any terminal and Eve have erasure probability $\delta = 0.5$.

In the initial phase, the terminals create shared information by exchanging packets. In the first round, Alice transmits $N = 8$ x-packets, $a_1, a_2, \ldots a_8$, of which Bob, Calvin, and Eve receive (not the same) half. Similarly, in the second and third rounds, Bob transmits $b_1, b_2, \ldots b_8$, and Calvin transmits $c_1, c_2, \ldots c_8$. Alice, Bob, and Calvin know which x-packets are received by one another (thanks to Step 2 of the initial phase), but not which x-packets are received by Eve.

Table II shows the x-packets known to each node at the end of the initial phase. Table III (top row) shows the x-packets shared by each terminal pair at the end of the initial phase (e.g., Alice and Bob share a_1, a_2, a_3, a_4 among others). To help visualize who knows which x-packets, from the x-packets shared by Alice/Bob, we mark those known to Eve as "canceled out" (e.g., $\not{a_3}$), those known to Calvin as "barred" (e.g., $\bar{a_2}$), and those known to both Eve and Calvin as both canceled out and barred (e.g., $\not{\bar{a_1}}$). We do the same for the other terminal pairs.

In the privacy amplification phase, the terminals create pairwise secrets by compressing their shared information.

Alice	Bob	Calvin	Eve
$a_1, a_2, \dots a_8$	a_1, a_2, a_3, a_4	a_1, a_2, a_5, a_6	a_1, a_3, a_5, a_7
b_1, b_2, b_3, b_4	$b_1, b_2, \dots b_8$	b_1, b_2, b_5, b_6	b_1, b_3, b_5, b_7
c_1, c_2, c_3, c_4	c_1, c_2, c_5, c_6	$c_1, c_2, \dots c_8$	c_1, c_3, c_5, c_7

TABLE II
INFORMATION KNOWN TO EACH NODE

Phase	Alice – Bob	Alice – Calvin	Bob – Calvin
Initial	$\cancel{a_1}, \cancel{a_2}, \cancel{a_3}, a_4$ $\cancel{b_1}, \cancel{b_2}, \cancel{b_3}, b_4$ $\cancel{c_1}, \cancel{c_2}$	$\cancel{a_1}, \cancel{a_2}, \cancel{a_5}, a_6$ $\cancel{b_1}, \cancel{b_2}$ $\cancel{c_1}, \cancel{c_2}, \cancel{c_5}, c_4$	$\cancel{a_1}, \cancel{a_2}$ $\cancel{b_1}, \cancel{b_2}, \cancel{b_5}, b_6$ $\cancel{c_1}, \cancel{c_2}, \cancel{c_5}, c_6$
Privacy Amp.	$a_3 + a_4$ $a_1 + a_2 + a_3$ $b_3 + b_4$ $b_1 + b_2 + b_3$	$a_5 + a_6$ $a_1 + a_2 + a_5$ $c_3 + c_4$ $c_1 + c_2 + c_3$	$b_5 + b_6$ $b_1 + b_2 + b_5$ $c_5 + c_6$ $c_1 + c_2 + c_5$

TABLE III
INFORMATION SHARED BY NODES

Alice and Bob compress their 10 shared x-packets into $M_{12} = 4$ shared y-packets (linear combinations of the shared x-packets). Similarly, Alice/Calvin and Bob/Calvin compress their 10 shared x-packets into 4 shared y-packets. Table III (bottom row) shows the y-packets shared by each terminal pair. Notice that Eve cannot reconstruct any of the y-packets. Moreover, Calvin cannot reconstruct the y-packets constructed by Alice and Bob for their pairwise secret. Similarly, Alice (Bob) cannot reconstruct the y-packets constructed by Bob (Alice) and Calvin for their pairwise secret.

This was an example to give a sense of how things work. Our protocol does not really construct so simple linear combinations (e.g., 5 random linear combinations out of 10 x-packets), as they may leak information to Eve (Section III-E).

D. Key Points

The size of the secret between two terminals depends on (1) the amount of information shared by the two terminals and (2) how much of this information Eve and the other terminals have missed. In the above example, Alice and Bob share 10 x-packets. Of these, Eve misses 5, and Calvin misses 4. Hence, Alice and Bob can construct up to 5 y-packets (linear combinations of their shared x-packets) that are perfectly secret from Eve, and up to 4 y-packets that are perfectly secret from Calvin. Since we want the Alice/Bob secret to be unknown to both Eve and Calvin, Alice/Bob should create only 4 y-packets. Creating a shorter secret would be inefficient. Creating a longer secret would necessarily result in Eve or Calvin knowing something about the secret (though not necessarily the entire secret).

An important feature of the protocol is that terminals T_i and T_j create shared information during all the rounds of the initial phase, not only when one of them transmits. In the above example, at the end of the initial phase, Alice and Bob share not only x-packets transmitted by one of them, but also x-packets transmitted by Calvin (c_1, c_2). In the particular example, these packets turn out not to be useful in creating the Alice/Bob secret, because Calvin knows both of them (and we want the secret to be unknown to Calvin). However, when we have more than $n = 3$ terminals, leveraging x-packets transmitted by all terminals becomes key to the protocol's scalability with the number of terminals. For instance, imagine that there is a fourth terminal, David, which transmits x-packets d_1, d_2, received by Alice/Bob, but not Calvin or Eve. Although d_1, d_2 are known to David, now Alice/Bob can create two combinations of c_1, c_2, d_1, d_2 (e.g., $c_1 + d_1, c_2 + d_2$) and create two extra y-packets unknown to Calvin, David, and Eve.

E. Secret Construction

Terminals T_i and T_j construct the following number of y-packets in the privacy amplification phase:

$$M_{ij} = \min \{ V_E, V_1, V_2, \dots V_n \}, \quad (1)$$

where:

- V_E is the expected number of x-packets that are shared by terminals T_i/T_j and missed by Eve.
- V_l is the number of x-packets shared by terminals T_i/T_j and missed by terminal T_l.

We compute V_E as $\sum_{k=1}^{n} U_{Ek}$, where $U_{Ek} = \delta_{kE} \cdot U_k$, and U_k is the number of x-packets transmitted by terminal T_k and received by both terminals T_i/T_j in round k of the initial phase. In short, we count, for each terminal and for Eve, how many of T_i/T_j's shared x-packets this terminal/Eve has missed (or is expected to have missed, in Eve's case), and we set M_{ij} to the smallest of these numbers.

It is straightforward to adapt this computation to the scenario where up to some number of terminals collude to learn $\mathcal{S}_{ij}$, but we do not consider this scenario in this paper.

Terminals T_i and T_j construct the y-packets using simple constructions based on Maximum Distance Separable (MDS) codes [7], as described in Lemma 5 in the Appendix. There is no novelty in these constructions (they rely on standard properties of MDS codes). One such property is that, if Eve has t packets, then each y-packet involves at least $t+1$ packets, which ensures that Eve cannot reconstruct it.

F. Active Adversaries and Authentication

To protect against active adversaries, the terminals need to share some bootstrap information to authenticate each other when they first communicate. Authentication is orthogonal to our secret agreement and can happen in different ways, e.g., by requiring the terminals to initially share bootstrap information and use it to construct authentication codes for the x-packets they transmit the first time they run our protocol. After the terminals have established their first pairwise secrets using our protocol, they can use these new secrets to construct new authentication codes, which do not depend on the bootstrap information.

One might argue: if the terminals have to share bootstrap information anyway to defend against active adversaries, they might as well share pairwise secrets to begin with and not run our protocol at all. The advantage of our protocol is that it enables the terminals to keep generating *new* secrets, independent from the previous ones, and continuously refresh their encryption and authentication keys. Unless the adversary can break into one of the terminals *while they run our protocol*, she has a small window of opportunity to compromise their

communication: she has to steal the bootstrap information and impersonate a terminal *while the terminals are running our protocol for the first time*.

IV. Protocol Analysis

In this section, we state certain properties of the basic protocol and also present an argument on why this particular protocol outperforms a more obvious alternative. We summarize the proofs of Lemmas 1 and 4 in the Appendix, Section A. We omit the proof of Lemma 2, which is straightforward.

Lemma 1. *If the theoretical network conditions hold, there exists a sufficiently large N for which the basic protocol is information-theoretically secure against a passive adversary.*

From the previous lemma, our protocol is secure; next we examine what efficiency it can achieve. Note that while for $n = 2$, we create a single key $\mathcal{S}$ with some efficiency E, for $n \geq 3$, the efficiency is different for each secret $\mathcal{S}_{ij}$, and depends on the erasure probabilities δ_{ki}, δ_{kj}, and δ_{kE}. In our notation, the efficiency simply corresponds to the ratio

$$E_{ij} = \frac{M_{ij}}{Nn}.$$

To calculate it, we need M_{ij}, to count how many packets a queue contains that Eve (or eavesdropping terminals) have not received. Over the theoretical network conditions, we can estimate M_{ij} using expected values. Lemma 6 in the Appendix provides concentration results showing that our estimation error becomes zero exponentially fast in the number of packets N. Lemma 2 provides such an example calculation.

Lemma 2. *If the theoretical network conditions hold, and we assume non-colluding eavesdroppers, then there exists a sufficiently large N for which the basic protocol achieves:*

- *$n = 2$ terminals, $E = \delta_E(1-\delta)$,*
- *$n \geq 3$, if $\delta_1 \leq \delta_{ij} \leq \delta_2 \ \forall\ i, j$ and $\delta_E = \min_i \delta_{iE}$,*

$$E_{ij} \geq \min\left\{ \delta_E(1-\delta_2)\left[(1-\delta_2) + \frac{2\delta_2}{n}\right], \right.$$
$$\left. \delta_1(1-\delta_2)\left[(1-\delta_2) - \frac{1-3\delta_2}{n}\right] \right\}.$$

This lemma verifies an intuitive fact: as the number of terminals (and transmission rounds in the initial phase) n increases, what dominates the size of each queue is the number of packets $(1-\delta_2)^2 N$ jointly overhead by two terminals; the fraction of these (δ_1 or δ_E) that is unknown to our strongest eavesdropper equals the amount of secrecy we can create. In other words, the fact that we keep adding x-packets in each queue during all rounds is the key in the protocol's scalability.

Lemma 3. *Under the conditions of Lemma 2, for $n = 2$ terminals, the basic protocol achieves maximum efficiency.*

Indeed, the efficiency we achieve for $n = 2$ reaches Maurer's upper bound [3].

The basic protocol scales well with the number of terminals because we try to leverage broadcasting as much as possible. If we were, instead, attempting pairwise secret establishment, the efficiency would quickly go to 0 with the number of terminals. To see this, consider the following, conceptually simpler alternative to the basic protocol: Consider a time-division protocol, where we operate in time-slots, and at each time-slot we create the key $\mathcal{S}_{ij}$ between a specific terminal pair, using the best possible protocol that achieves efficiency $\delta_E(1-\delta)$ [3]. Since we have $\binom{n}{2}$ keys to create, and each key is created during only one time-slot, the overall efficiency is $E^{(\text{alt})} = \frac{\delta_E(1-\delta)}{\binom{n}{2}}$ per key. Unlike the efficiency of our protocol that converges to a constant value as n increases, $E^{(\text{alt})}$ goes to zero.

Finally, the most demanding operations a terminal needs to perform is linear combining to create the y-packets. Thus:

Lemma 4. *Each terminal that participates in the basic protocol executes an algorithm that is polynomial in N and n.*

V. Adapting to Real Networks

In this section, we adapt our basic secret-agreement protocol (Section III) to the scenario where the theoretical network conditions do not hold.

A. Basic Idea

The challenge with real networks is that we do not know the size of the pairwise secrets (the M_{ij} from Section III) that we should create. In Section III-E, we were able to analytically compute M_{ij} because we assumed that we knew enough about Eve's channels to compute the expected amount of information missed by Eve.

We conservatively estimate the amount of information missed by Eve based on the amount of information missed by the terminals. More specifically, Alice and Bob assume that, during each round of the initial phase, Eve learns as much information as any of the other terminals about the x-packets shared by Alice/Bob. Hence, at the end of the initial phase, Eve is assumed to know at least as many of the Alice/Bob shared x-packets as the most knowledgeable terminal.

We chose this based on the following observations: Channel behavior varies significantly over time, to the point where we cannot estimate or even upper-bound how much information Eve collects during one experiment based on how much information she collected during past experiments. Channel behavior also varies over *space*, but less so: if, during an experiment, terminal T_i receives many packets in common with neighbor T_j, then T_i most likely receives many packets in common with its other neighbors as well. It turns out that, by measuring how many packets each pair of neighboring terminals receive in common during one experiment, we can estimate quite accurately how many packets any terminal and Eve receive in common *in the same experiment*. This, of course, is an empirical estimation, thus we cannot guarantee its accuracy theoretically.

B. Secret Construction

Terminals T_i and T_j estimate that, at the end of the initial phase, from their shared x-packets, Eve misses the following

number:

$$V_E = \sum_{k \neq i,j,\ k=1}^{n} \min\{\ V_1^k, V_2^k, \ldots V_n^k\ \}, \qquad (2)$$

where:

- V_l^k is the number of new x-packets shared by terminals T_i/T_j and missed by terminal T_l during round k of the initial phase.

In short, we assume that, in each round of the initial phase, Eve missed as few (of the x-packets newly shared by T_i/T_j in this round) as any other terminal.

C. Key Points

If we do not assume theoretical network conditions, we cannot offer formal guarantees about the reliability of our protocol, because we do not know exactly how much information Eve collects during the initial phase: it is theoretically possible that Eve receives more x-packets in common with the terminals than we estimate, which means that she learns something about the pairwise secrets. The amount of information that leaks to Eve depends both on the particular wireless network and the number of terminals: the more terminals we have, the more we learn about the quality of channels throughout the network, and the better we can estimate the quality of Eve's channels. Hence, the amount of information that leaks to Eve needs to be experimentally assessed in each wireless network as a function of the number of terminals n.

VI. Experimental Evaluation

In this section, we experimentally evaluate our adapted secret-agreement protocol (Section V) on a small wireless testbed. Our goal is to answer two questions: is it feasible to achieve non-negligible secrecy rate in a real wireless network by leveraging packet erasures? and how well can we do so using our protocol?

A. Testbed

We show our testbed in Figure 1. It consists of 6 nodes distributed over an indoor office area. Unless otherwise specified, the nodes are HTC Wildfire Android smartphones. We set the phones to 802.11 ad-hoc mode, and we fixed their transmission rate to 36 Mbps. In some experiments, we also use WARP software radios [8].

In order for our approach to work, the wireless network must provide a certain level of channel variability. The simplest scenario where such variability exists is when the nodes are not in direct line of sight, e.g., they are separated by office walls. This is the scenario we implement in our testbed. Our protocol can work even when the nodes are in direct line of sight, but for that we need to use artificial noise (the terminals create interference and force Eve to miss some of the traffic they exchange). We are currently experimenting with that idea (we have some early results here [9]), but we do not consider this approach in this paper.

When we refer to an "experiment," we mean that we place one node in each room, and we run one round of our protocol.

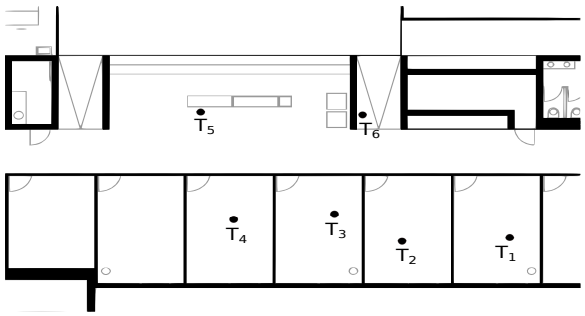

Fig. 1. Our testbed. Each office is about 2×3 meters.

In each experiment, one node plays the role of Eve, while the rest play the role of 5 terminals that exchange pairwise secrets. There are 6 possible arrangements of 5 terminals and Eve in 6 rooms, and we experiment with 4 different levels of transmit power. Hence, each presented graph summarizes the results of 4×6 experiments (all the combinations of transmit-power levels and node arrangements).

We present two kinds of graphs: minimum reliability as a function of transmit power and minimum efficiency/secrecy rate as a function of transmit power. Each reliability value is the minimum reliability achieved *by any terminal pair* in any of the 6 node arrangements. Each efficiency/secrecy rate value is the minimum efficiency/secrecy rate achieved by any terminal pair in any node arrangement.

B. Ideal Secrecy Rate

We start by looking at the ideal efficiency and secrecy rate achievable in this testbed by leveraging packet erasures. In particular, we measure the efficiency and secrecy rate of an *oracle-assisted* protocol; this works like ours, with the only difference that it does not estimate how much information Eve obtains in the initial phase—that knowledge is directly provided by the oracle. More specifically, instead of estimating V_E using Equation 2, we set it to the exact number of x-packets shared by terminals T_i/T_j and missed by Eve. This oracle-assisted protocol by construction achieves reliability 1, because it knows exactly how much information Eve obtains in the initial phase and computes the longest secret that is completely unknown to Eve. Figure 2 ("Ideal" label) shows the minimum efficiency/secrecy rate achieved by any terminal pair using the oracle-assisted protocol, as a function of the transmit power of the terminals.

First, we see that, if we perfectly knew Eve's channel conditions, each terminal pair could exchange tens of thousands of secret bits per second (up to 55 secret Kbps, achieved for a transmit power of 10 dbm), of which Eve would have zero information independently from her computational capabilities. We consider this an encouraging first result: It shows that a real wireless network may offer enough channel variability to enable the exchange of tens of thousands of secret bits per second—enough to create a brand new 256-bit encryption key every few milliseconds. Of course, we expect this secrecy rate to become smaller as we consider adversaries with increasingly more sophisticated hardware (e.g., multiple

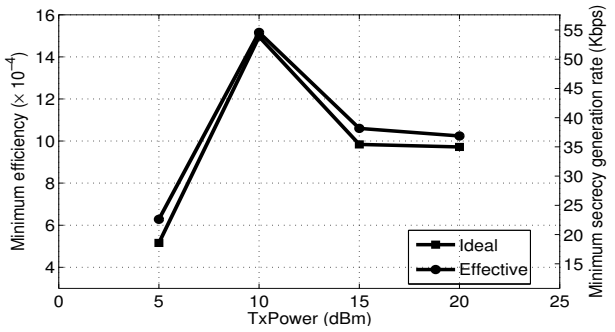

Fig. 2. Minimum efficiency and secrecy rate as a function of TX power. "Ideal" corresponds to the oracle-assisted protocol. "Effective" corresponds to our protocol. Both protocols create up to 55 secret Kbps.

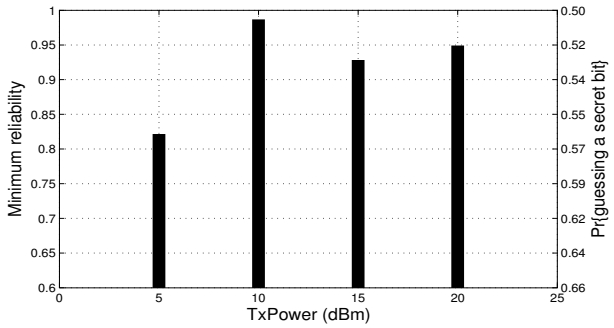

Fig. 3. Minimum reliability of our protocol as a function of TX power. All the pairwise secrets that we create have reliability above 0.8.

receiving antennas). However, this first result is good enough to give us hope that we can achieve non-negligible secrecy rate even against sophisticated adversaries.

Second, we see that secrecy rate first increases, then drops as the transmit power of the terminals increases. This is due to the following reason: As the transmit power of a terminal increases, so does the quality of its channels to both the other terminals and Eve. Hence, higher transmit power means that (1) the terminals exchange traffic faster, but also that (2) Eve overhears more of their exchanges. At low transmit power, effect 1 dominates, and increasing the transmit power improves secrecy rate; beyond 10 dbm, effect 2 dominates, and increasing the transmit power benefits Eve more than the terminals, decreasing the secrecy rate.

C. Reliability and Secrecy Rate of our Protocol

Next, we look at the performance of our protocol. Unlike the oracle-assisted protocol, ours needs to estimate how much information Eve obtains in the initial phase. If it overestimates Eve's knowledge, it creates a shorter secret than it could, achieving lower efficiency/secrecy rate than the oracle-assisted protocol. If it underestimates Eve's knowledge, it creates a longer secret than it should, achieving higher secrecy rate than the oracle-assisted protocol, but reliability below 1. Hence, there is a trade-off between secrecy rate (how fast we create new secrets) and reliability (how secure these secrets are).

Ideally, we would want our protocol to behave like the oracle-assisted one (achieve the same secrecy rate and reliability 1). In practice, this is infeasible, as it would require us to always estimate Eve's knowledge with perfect accuracy. Thankfully, it is also unnecessary: Suppose a secret has reliability 0.8, which means that Eve can correctly guess the value of one bit of the secret with probability $2^{-0.8} = 0.57$. The smallest secret that our protocol ever creates is one y-packet (1 KB); reliability 0.8 means that Eve can correctly guess the value of one y-packet with probability $2^{-0.8 \cdot 8000} \approx 0$. Hence, as long as the terminals use their pairwise secrets at the granularity of a y-packet (e.g., they use at least one entire y-packet as an encryption key), they are secure from Eve. Figure 2 ("Effective" label) and Figure 3 show the minimum efficiency/secrecy rate and minimum reliability of our protocol per constructed $\mathcal{S}_{ij}$ (where the minimum is over all i, j) as a function of the transmit power of the terminals.

Our protocol creates tens of thousands of secret bits per second between each terminal pair (up to 55 secret Kbps, achieved for a transmit power of 10 dbm). All of the pairwise secrets that we create have reliability above 0.8 (and most of them have reliability 1). We consider this an encouraging result, too: It shows that, in a real wireless network, it may be feasible to accurately estimate an adversary's knowledge, if we have a sufficiently dense deployment of collaborating honest nodes. Of course, this estimation will become harder as we consider adversaries with increasingly more sophisticated hardware. However, this first result is good enough to give us hope that we can achieve reasonable accuracy even against sophisticated adversaries.

Our next step. In the above experiments, Eve gains knowledge only from the information that is successfully delivered to her application layer. There exist packets that reach Eve's receiver, yet are not delivered to her application because they are corrupted beyond what the lower layers can repair. One could argue that, if Eve rooted her phone and gained access to every packet that reaches her physical layer, she would improve her knowledge. The question is by how much.

To start answering this question, we used three WARP software-defined radios[3], configured with an 802.11-compliant physical layer (16 QAM over OFDM), and we placed them in our testbed. We make one of them (Alice) send out traffic, while the other two (Bob and Eve) receive. The difference from our earlier experiments is that now Eve tries to use *all* the packets that reach her physical layer (with corrupted payload or not) to increase her knowledge. First, we consider the packets that are correctly received by Bob, and we measure Eve's knowledge (conditional entropy) about these packets[4]. Then we repeat the experiment, assuming that an oracle magically repairs the corrupted payload of every packet that reaches Eve's receiver. In the former case, Eve's uncertainty on Bob's information originates from both corrupted and erased symbols, whereas in the latter only from erased ones (that do not reach Eve's receiver at all).

[3]The smartphones used for our earlier experiments do not provide access to received data that is discarded below the application layer.

[4]We do so by calculating the joint empirical distribution of the 16-QAM symbols in the payload of Bob's packets and the symbols that Eve receives—correctly or not.

The results show that—at least in our testbed—Eve's uncertainty mostly depends on the erased in-the-air symbols, i.e., symbols that were not demodulated at all. For instance, if Alice uses a transmit power of 15 dbm, in the second experiment (where all payload corruption is corrected by the oracle), Eve learns only an extra 0.1 bit-per-channel-use relative to the first experiment. This indicates that the number of partially corrupted packets that reach Eve's receiver is relatively small, hence they do not significantly increase Eve's knowledge (or reduce the secrecy rate achieved by our protocol).

We are currently examining more sophisticated receivers that work directly with the analog physical signal (rather than at symbol level), to examine ultimate information leakage.

VII. Related Work

There exists a rich literature on creating shared secrets between two nodes by leveraging channel variations, which can be broadly divided into two categories: On the one hand, there exists theoretical work on creating unconditionally secure secrets under idealized conditions. The seminal contributions were by Wyner [2] and Maurer [3] (see also [10] and references therein). On the other hand, there exists practical work that presents concrete, implementable protocols for wireless environments (for lack of space, we only mention those that were evaluated through implemented prototypes). Some of them leverage the time-varying nature and reciprocity of wireless channels [11], [12], [13]. These achieve secret-generation rates up to a few tens of bps (in modified 802.11 or 802.15 environments). In another proposal, Alice and Bob create shared secrets by combining and heuristically condensing the frames that are transmitted between them only once (based on the assumption that these frames are less likely to have been overheard by Eve) [14]. This has been implemented in an 802.11 environment, but has not been evaluated in terms of efficiency or reliability yet. In iJam, when Alice transmits, Bob jams a part of her transmission in a special way (specific to OFDM) that prevents Eve from guessing which part was jammed; hence, Alice and Bob share common knowledge that is secret from Eve, and they use it to create shared secrets [15]. This achieves a secret-generation rate up to 18 Kbps (in a modified 802.11 environment).

We differ in the following ways: To the best of our knowledge, our work is the first to consider *multi-terminal* pairwise secret agreement, where broadcast is leveraged to efficiently create multiple shared secrets at the same time. The existing protocols focus on a single pair of nodes, hence they are not designed to leverage broadcast, and they would not scale well with the number of terminals (if applied to the multi-terminal scenario). Moreover, our protocol achieves a secret-generation rate of tens of Kbps, without requiring any custom physical-layer operations that are specific to OFDM (or any other transmission scheme). Finally, we should mention our earlier workshop paper, which also presents a protocol for creating shared wireless secrets, although that focuses on group (not pairwise) secrets and relies on artificial interference [16].

VIII. Discussion and Conclusions

We have presented a protocol that enables a group of terminals, connected to the same broadcast domain, to exchange pairwise secrets in the presence of an adversary. We assume nothing about the adversary's computational capabilities, but we assume that she cannot overhear *all* the information received by any terminal. The key properties that differentiate our protocol from related theoretical work are that it has polynomial complexity, is readily implementable in simple wireless devices, and scales well to an arbitrary number of terminals. On the practical side, we deployed our protocol on a small wireless testbed, where the terminals and the adversary are smartphones located in adjacent offices. We presented experimental evidence that 5 terminals can create pairwise secrets at a rate of 55 Kbps, with their secrecy being independent from the adversary's computational capabilities.

We developed our protocol under the assumption that the terminals are "honest but curious." With a bit extra care, we can relax it: The worst a dishonest terminal T_i can do is lie about which packets it misses in the initial phase to cause the other terminals to overestimate the number of packets missed by Eve and create unreliable secrets. This works, only if T_i is the terminal that missed the fewest packets in every round of the initial phase. We can further increase our robustness by estimating the secret size even more conservatively, e.g., by assuming that, in each round, Eve missed a fraction of the second smallest value reported by any terminal.

Our biggest challenge will be to make our protocol robust against an adversary that possesses multiple receiving antennas. Clearly, the more antennas an adversary has, the more information she can receive. We need to tune our conservativeness about the created secret size as a function of the strength of the adversary that we want to thwart. For instance, if we assume that Eve can be present at two separate network locations, then Alice and Bob should estimate how much of their shared information is missed by Eve based on how much information is missed by pairs of terminals (as opposed to single terminals). Our ultimate goal is to determine the practical limits of our approach—how does secrecy rate decrease as the network presence of the adversary increases.

References

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[2] A. D. Wyner, "The Wire-tap Channel," *Bell System Tech. J.*, vol. 54, pp. 1355-1387, 1975.

[3] U. M. Maurer, "Secret Key Agreement by Public Discussion from Common Information," *IEEE Transactions on Information Theory*, vol. 39, pp. 733-742, 1993.

[4] Companies Developing QKD Systems: Id Quantique, MagiQ Technologies, SmartQuantum, Quintessence Labs.

[5] A. Mink, X. Tang, L. Ma, T. Nakassis, B. Hershman, J.C. Bienfang, D. Su, R. Boisvert, C. W. Clark, and C. J. Williams, "High Speed Quantum Key Distribution System Supports One-time Pad Encryption of Real-time Video," in *Proceedings of SPIE*, vol. 6244, 2006.

[6] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, and A. Markopoulou, "Microcast: Cooperative Video Streaming on Smartphones," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2012.
[7] F. J. Macwilliams and N. J. A. Sloane, "The Theory of Error Correcting Codes," *North-Holland*, 2006.
[8] "Rice University Wireless Open-Access Research Platform (WARP)," http://warp.rice.edu.
[9] M. Jafari, U. Pulleti, E. Atsan, I. Safaka, C. Fragouli, K. Argyraki, and S. Diggavi, "Exchanging Secrets Without Using Cryptography", http://arxiv.org/abs/1105.4991.
[10] B. Kanukurthi and L. Reyzin, "Key Agreement from Close Secrets over Unsecured Channels," in *Proceedings of the ACM/USENIX EUROCRYPT Conference*, 2009.
[11] B. Azimi-Sadjadi, A. Kiayias, A. Mercado, and B. Yener, "Robust Key Generation from Signal Envelopes in Wireless Networks," in *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2007.
[12] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. Mandayam, "Information-theoretically Secret Key Generation for Fading Wireless Channels," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 240-254, 2010.
[13] J. Croft, N. Patwari, and S. Kasera, "Robust Uncorrelated Bit Extraction Methodologies for Wireless Sensors," in *Proceedings of the ACM/IEEE Conference on Information Processing in Sensor Networks (IPSN)*, 2010.
[14] S. Xiao, W. Gong, and D. Towsley, "Secure Wireless Communication with Dynamic Secrets," in *Proceedings of the IEEE INFOCOM Conference*, 2010.
[15] S. Gollakota and D. Katabi, "Physical Layer Wireless Security Made Fast and Channel Independent," in *Proceedings of the IEEE INFOCOM Conference*, 2011.
[16] I. Safaka, C. Fragouli, K. Argyraki, and S. Diggavi, "Creating Shared Secrets Out of Thin Air," in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2012.
[17] M. Mitzenmacher and E. Upfal, "Probability and Computing, Randomized Algorithm and Probabilistic Analysis," *Cambridge University Press*, 2006.

APPENDIX

A. Proof of Lemma 1

We consider our eavesdropper to be either Eve or one of the participating terminals. Our eavesdropper has two occasions to obtain information about the secret $\mathcal{S}_{ij}$: by overhearing a fraction of the transmitted x-packets in the initial phase; or because a terminal knows the source packets it transmitted. Both these effects are captured in the calculation of the number M_{ij}. Under the theoretical network model conditions, we can approximate these numbers with their average value; Lemma 6 shows that this approximation can become arbitrarily good exponentially fast in N. Given that we use any value M_{ij} smaller or equal to the exact, the following Lemma 5 gives a construction that does not allow the eavesdropper to obtain any information about $\mathcal{S}_{ij}$. ■

Lemma 5. *Consider a set of N x-packets, say $x_1, \ldots, x_N$, and assume an eavesdropper, Eve, has a subset of size N_E of the x-packets. Construct $M = N - N_E$ y-packets, say $y_1, \ldots, y_M$, as*

$$Y = AX,$$

*where matrix X has as rows the N x-packets, matrix Y has as rows the $N - N_E$ y-packets, and A is the generator matrix of a Maximum Distance Separable (MDS) linear code with parameters $[N, N - N_E, N_E + 1]$ (*e.g.*, a Reed-Solomon code [7]). Then the M y-packets are information-theoretically secure from Eve, irrespective of which subset (of size N_E) of the x-packets Eve has.*

Proof: ▷Let W be a matrix that has as rows the packets Eve has. To prove that the y-packets are information-theoretically secure from Eve, we must show that:

$$H(Y|W) = H(Y).$$

▷ We can write

$$\left[\begin{array}{c} Y \\ W \end{array} \right] = \left[\begin{array}{c} A \\ A_E \end{array} \right] X \stackrel{def}{=} BX,$$

where A_E is a $N_E \times N$ matrix of $\mathrm{rank}(A_E) = N_E$, which specifies the N_E distinct x-packets that are known to Eve. A_E is *not known* to us, however we know is that in each row of A_E there is only one 1 and the remaining elements are zero; so all of the vectors in the row span of A_E have Hamming weight (the number of nonzero elements of a vector [7]) less than or equal to N_E. On the other hand, from construction, $\mathrm{rank}(A) = N - N_E$, and each vector in the row span of A has Hamming weight larger than or equal to $N_E + 1$ [7]; thus the row span of A and A_E are disjoint (except for the zero vector) and the matrix B is full-rank, i.e. $\mathrm{rank}\,(B) = N$.

▷ If the packets x_i have length Λ, we have that:

$$\begin{aligned} &H(Y|W) = H(Y,W) - H(W) = \\ &= \mathrm{rank}\,(B)\,\Lambda - \mathrm{rank}(A_E)\Lambda = (N - N_E)\Lambda \\ &= \mathrm{rank}(A)\Lambda = H(Y). \end{aligned}$$ ■

B. Concentration to expected values

Lemma 6. *The values of the random variables M_{ij}, U_{kE}, V_l, as defined in Section III-E and used in Lemma 1, converge exponentially fast in N to their expected values.*

Proof: Consider the random variable U_{kE} denoting the number of x-packets transmitted by T_k and received by both T_i/T_j but not Eve. We use a standard argument to show it concentrates exponentially fast to its average. Define the random variable $\eta_q^{(l)}$ as

$$\eta_q^{(l)} = \begin{cases} 1 & \text{if the } q\text{th } x\text{-packet is received} \\ & \text{by terminals } T_i/T_j \text{ and missed by Eve,} \\ 0 & \text{otherwise.} \end{cases}$$

Then we can write $U_{kE} = \sum_{q=1}^{N} \eta_q^{(l)}$ and we have

$$\mu \triangleq E(U_{kE}) = (1 - \delta_{ki})(1 - \delta_{kj})\delta_{kE} N.$$

For $0 < \epsilon \leq 1$ we can write $\mathbb{P}\left[U_{kE} - \mu \geq \epsilon\mu\right] \leq \exp\left(-\frac{\epsilon^2\mu}{3}\right)$, where in the last inequality we use Chernoff bound [17, Chapter 4]. We can also write, for $0 < \epsilon \leq 1$, $\mathbb{P}\left[U_{kE} - \mu \leq \epsilon\mu\right] \leq \exp\left(-\frac{\epsilon^2\mu}{2}\right)$. Similar arguments hold for the remaining variables in Section III-E. ■