Creating Shared Secrets out of Thin Air

Iris Safaka[*], Christina Fragouli, Katerina Argyraki,
Computer and Communication Sciences, EPFL, Switzerland
{iris.safaka, christina.fragouli, katerina.argyraki}@epfl.ch

Suhas Diggavi[†]
Electrical Engineering, UCLA, CA, USA
suhasdiggavi@ucla.edu

ABSTRACT

Current security systems typically rely on the adversary's computational limitations (e.g., the fact that it cannot invert a hash function or perform 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 its limited network presence (i.e., that the adversary cannot be located at many different points in the network at the same time). We take a first step toward designing and building a wireless security system that leverages this opportunity: We consider the problem where a group of n nodes, connected to the same broadcast wireless network, want to agree on a shared secret (e.g., an encryption key), in the presence of an adversary Eve who tries to listen in and steal the secret. We propose a secret-agreement protocol, where the n nodes of the group keep exchanging bits until they have all agreed on a bit sequence that Eve cannot reconstruct (with very high probability). We provide experimental evidence—to the best of our knowledge, the first one—that a group of wireless nodes can generate thousands of new shared secret bits per second, with their secrecy being independent of the adversary's computational capabilities.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Security

Keywords

Information-theoretic security, group secrets

[*]Supported by ERC Starting Grant ERC-2009-StG-240317

[†]Partly supported by AFOSR MURI, prime award FA9550-09-064

Hotnets '12, October 29–30, 2012, Seattle, WA, USA.

1. INTRODUCTION

One of the most fundamental problems in system security is creating shared secrets (without using an out-of-band channel), and the only way we know to solve this problem today relies on the adversary's computational limitations. Consider two communicating principals, Alice and Bob, and an adversary Eve, who is eavesdropping on their channel; how can Alice and Bob create a shared secret such that Eve obtains very little information on the secret? One typically assumes that Alice and Bob share some initial piece of information (e.g., each other's public key) and use asymmetric secret-agreement algorithms, like RSA [1], which fundamentally assume that Eve is technologically limited in her ability to perform certain computations, like large-integer factorization, in useful time.

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. If Alice is a wireless node, when she transmits, she may be overheard by many receivers, however, as long as there is sufficient noise, it is unlikely that any two receivers overhear exactly the same information. To illustrate, consider a person speaking in a low voice in a noisy room: people standing nearby will hear parts of the speech, but if the room is sufficiently noisy, it is unlikely that any two of these people (standing at different locations) will hear the exact same parts of what the speaker says. 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].

Turning this feasibility result into a practical secret-agreement protocol can have significant implications for wireless security. It would enable Alice and Bob to generate shared secrets "out of thin air" (and use them, e.g., to continuously refresh the key used to encrypt their communication [4]). These continuously generated shared secrets would not rely on any information permanently stored in Alice's or Bob's machines. For instance, there would be no public/private RSA key pair or master key (as in WPA) that, if stolen or accidentally revealed, would enable an adversary to reconstruct Alice's and Bob's shared secrets (and decrypt their communication).

We do not advocate replacing existing crypto-systems that rely on the adversary's computational limitations. However, we believe that exploring complementary 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 industry, where several companies are developing quantum key distribution (QKD) systems; such systems enable a pair of nodes to exchange a secret such that an eavesdropper obtains very little information on the secret, independently from her computational capabilities. A typical application envisioned for QKD 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 expensive (due to the need for sophisticated equipment such as photon detectors), hence 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 consider a set of n wireless nodes (we will call them "terminals"), connected to the same broadcast network, and an eavesdropper Eve connected to the same network. We design a protocol that allows the terminals to create a shared secret such that Eve learns very little about the secret. If Eve is a passive adversary (she never makes any transmissions), the terminals do not need to share any information before they run our protocol. If Eve is an active adversary (hence may try to impersonate a terminal), then the terminals need to share a (small) initial piece of information when they first communicate (until they generate their first shared secret). 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 shared secret. However, any shared secrets subsequently generated through the protocol do not depend in any way on the bootstrap information.

Relative to actual security systems, we shift the challenge from computation to network presence: for current crypto-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. Which of these hurdles is more challenging for an adversary depends on the system and setup, and this guides the system designer in making the choice. Or, if defense in depth is desired, both forms of protection could be employed simultaneously.

We build on ideas from the information theory community and develop an actual protocol of polynomial complexity that is implementable in simple wireless devices. We also advance the theory state of the art by focusing on the creation of *group* secrets (as opposed to secrets between pairs of nodes). This choice is motivated by the increasing tendency of wireless users to consume content in groups [6], enabled by the accessibility of social applications on mobile phones.

An important difference from theoretical work is that we do not assume that the terminals know the exact quality of the channel between each terminal and Eve. This assumption is necessary for formally proving "information theoretic security," i.e., that Eve will obtain zero information about the secret, but we found that this does not always hold in wireless networks. We therefore use artificially generated interference [7, 8] to ensure that Eve, wherever she is located, will miss some minimum fraction of the information transmitted by any terminal with very high probability.

As a proof of concept, we deployed our protocol on a small wireless testbed that covers an area of 14 m^2 and consists of $n = 8$ terminals, 6 interferers, and an adversary. Our experiments so far show that, in our testbed, if the adversary uses a standard wireless physical layer and is located within no less than 1.75 m from any terminal, the $n = 8$ terminals achieve a secret generation rate of 38 kilobits per second. To the best of our knowledge, this is the first experimental evidence that a group of wireless nodes can generate thousands of new shared secret bits per second without relying on the adversary's computational limitations.

After introducing our setup (§2), we describe our protocol (§3), our deployment and early experimental results (§4), and related work (§5). We close with the hardest challenge that remains to be solved before our protocol is ready for use in practice (§6).

2. SETUP

We consider n wireless nodes, $T_0, \ldots, T_{n-1}$, connected to the same broadcast network; we will refer to these nodes as *terminals*; sometimes we will refer to terminals T_0, T_1, and T_2 respectively as Alice, Bob, and Calvin.

We consider an adversary, Eve, connected to the same broadcast network as the terminals. Our design assumes that Eve may possess multiple receiving antennas. But we should state upfront that the experimental results presented in Section 4 assume that Eve is a standard 802.11 wireless router with one omnidirectional antenna.

The terminals communicate with each other in two ways: (1) When we say that terminal T_i *transmits* a packet, we mean that it broadcasts the packet once. (2) When we say that terminal T_i *reliably broadcasts* a packet, we mean that it ensures that all other terminals T_i receive it, e.g., through acknowledgments and retransmissions; to be conservative, we assume that Eve receives all reliably broadcast packets.

Our goal is to design a protocol that enables the n terminals to create a shared secret $\mathcal{S}$, in a way that Eve obtains very little information on $\mathcal{S}$. We assume that Eve is a passive adversary, i.e., she does not perform any transmissions, she only tries to eavesdrop on the communications. We describe how to defend against active attacks (through authentication) in our technical report [9].

3. PROTOCOL

3.1 Phase 1: Pair-wise Secrets

Basic idea. Suppose Alice and Bob exchange (and thus know the contents of) 3 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 x_1 and x_2, they could simply use $\langle x_1, x_2 \rangle$ as their shared secret. If an oracle told Alice and Bob that Eve misses at least two of their shared packets (but not which two), they could still create a perfect shared secret (one that Eve knows nothing about), by using two linear combinations of their shared packets, e.g., $\langle x_1 \oplus x_2, x_2 \oplus x_3 \rangle$ (where $\oplus$ denotes the bit-wise XOR operation over the payloads of the corresponding packets).

Algorithm.

1. Alice (T_0) transmits N packets (we will call them x-*packets*).

2. Each terminal $T_{i \neq 0}$ reliably broadcasts the identities of the x-packets it received correctly.

3. Alice constructs M linear combinations of the x-packets (we will call them y-*packets*), using a well-defined construction (specified in [9], due to space restrictions). She reliably broadcasts the identities of the x-packets she used to create each y-packet.

4. Each terminal $T_{i \neq 0}$ reconstructs the contents of as many (say M_i) of the y-packets as it can based on the x-packets it received.

At this point, Alice and terminal T_i share M_i y-packets. Their shared pair-wise secret is the concatenation of these packets.

Example. Suppose we have $n = 2$ terminals, Alice and Bob. First, Alice and Bob create some shared information between them: Alice transmits $N = 10$ x-packets, $x_1, x_2, \ldots, x_{10}$ (step 1). Bob correctly receives 5 of them, x_1, x_3, x_5, x_7, x_9, and tells Alice which ones (step 2). Suppose Eve correctly receives 6 of the transmitted packets, $x_1, x_3, x_5, x_6, x_8, x_{10}$, and completely misses the rest. At this point, Alice and Bob share the contents of x_1, x_3, x_5, x_7, x_9; of these, Eve misses x_7, x_9.

Next, Alice and Bob perform privacy amplification, i.e., they condense their 5 shared x-packets into $M_1 = 2$ shared y-packets: Alice constructs two y-packets that are linear combinations of the x-packets she shares with Bob: $y_1 = x_1 \oplus x_5 \oplus x_9$ and $y_2 = x_3 \oplus x_7$. Then, Alice reliably broadcasts the identities of the x-packets she used to construct the y-packets, but not their contents (step 3). Bob uses this information to reconstruct the contents of the y-packets (step 4). Eve overhears Alice's reliable broadcast, but she cannot reconstruct the contents of either y_1 or y_2, because she does not know the contents of x_7 or x_9. At this point, Alice and Bob share y_1, y_2, whereas Eve knows nothing about them.

It is important that Alice construct the y-packets using a particular construction, because not any linear combinations of x-packets will do. The y-packets that we use above happen to work for the particular example, but our protocol does not really construct so simple linear combinations, as they may leak information to Eve. For instance, suppose Alice constructed the y-packets as follows: $y'_1 = x_1 \oplus x_3 \oplus x_5$ and $y'_2 = x_7 \oplus x_9$. In this case, Eve would be able to reconstruct y'_1, hence recover half of the shared secret.

Key point. In this example, Eve knows nothing about the shared secret $\langle y_1, y_2 \rangle$, because it consists of 2 linear combinations of the x-packets, which is the number of x-packets shared by Alice and Bob but not Eve at the end of step 2. So, to create a perfect shared secret with terminal T_i, Alice needs to know a lower bound for the number of x-packets shared with T_i that Eve has missed, and she must set M_i (the size of the pair-wise secret shared with T_i) to this value. The closer this lower bound is to the actual number of x-packets that Eve has missed, the longer the shared pair-wise secret. A key question is: how can Alice estimate a good lower bound in practice?

3.2 Phase 2: Group Secret

Basic idea. Once Alice has created a perfect pair-wise secret with each terminal T_i, she could use this secret to unicast a group secret to T_i. This "unicast" algorithm, however, has poor scalability: It requires Alice to make $n - 1$ separate transmissions. As a result, its efficiency (the size of the group secret divided by the minimum amount of data that Alice needs to transmit to create the group secret) goes to 0 as the number of terminals n increases. This means that the terminals cannot create any group secret at all.

Instead, we use the following algorithm: Alice transmits a second round of packets, which does not increase the number of secret bits shared by Alice and each other terminal, but "redistributes" them, such that all terminals share the same secret bits. The size of the resulting shared group secret is equal to the size of the shortest pair-wise secret.

Figure 1 shows the efficiency of the unicast algorithm (dashed lines) as well as our algorithm (continuous lines) for different values of n, under simplifying assumptions (Alice guesses exactly the number of x-packets shared with terminal T_i that are missed by Eve; the packet erasure probability between Alice and each terminal, as well as Alice an Eve, is the same). Even under these (favorable) assumptions, the efficiency of the unicast algorithm goes to 0 as n increases.

Algorithm.

1. Alice (T_0) constructs $M - \min\{M_1, M_2, \ldots M_{n-1}\}$ linear combinations of the y-packets (we will call them z-*packets*), using a well-defined construction [9]. She reliably broadcasts both the contents of each z-packet and the identities of the y-packets used to construct each z-packet.

2. Each terminal $T_{i \neq 0}$ reconstructs the $M - M_i$ y-packets it is missing by combining any of the $M - M_i$ z-packets with the M_i y-packets it reconstructed in step 4 of phase 1.

3. Alice constructs $L = \min\{M_1, M_2, \ldots M_{n-1}\}$ linear

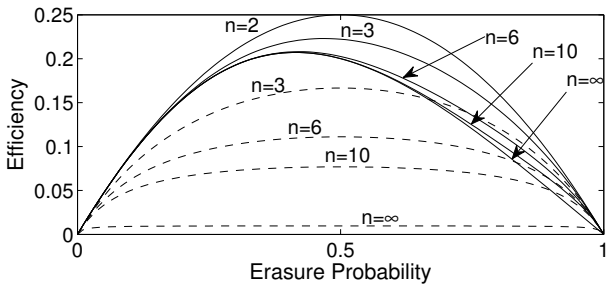

Figure 1: Maximum efficiency of our algorithm (continuous lines) and the unicast algorithm (dashed lines).

combinations of the y-packets (we will call them *s-packets*), using a well-defined construction [9]. She reliably broadcasts the identities of the y-packets that she used to create each s-packet.

4. Each terminal is now able to reconstruct all the s-packets, because it has all the y-packets.

At this point, all terminals have the same set of L s-packets. The shared group secret is the concatenation of these L shared s-packets.

Example. Suppose we have three terminals, Alice, Bob, and Calvin. At the end of phase 1, Alice has constructed $M = 3$ y-packets. Of these, she shares $M_1 = 2$ with Bob (y_1 and y_2) and $M_2 = 2$ with Calvin (y_1 and y_3). Suppose Eve knows nothing about any of the y-packets.

First, we reach a point where all terminals share all the y-packets: Alice constructs $M - min\{M_1, M_2\} = 1$ linear combination of the y-packets, $y_2 \oplus y_3$, and reliably broadcasts its contents (step 1). Thanks to this information, each of Bob and Calvin reconstructs the y-packet that he is missing (step 2). At this point, all terminals share y_1, y_2, y_3, while Eve knows the value of $y_2 \oplus y_3$.

Next, the terminals perform privacy amplification, i.e., they condense their 3 shared y-packets into $L = min\{M_1, M_2\} = 2$ shared s-packets: Alice constructs 2 s-packets that are linear combinations of the y-packets: $s_1 = y_1 \oplus y_2 \oplus y_3$ and $s_2 = y_1 \oplus y_2$. Then, Alice reliably broadcasts the identities of the y-packets she used to create the s-packets, but not their contents (step 3). Bob and Calvin use this information to reconstruct the contents of the s-packets (step 4). Eve overhears Alice's reliable broadcast, but she cannot reconstruct the contents of s_1 or s_2, because she does not know the contents of y_1, y_2. At this point, Alice, Bob, and Calvin share s_1, s_2, whereas Eve knows nothing about them.

As with the y-packets, it is important that Alice construct the z-packets and s-packets using particular constructions, otherwise they may leak information to Eve.

Key point. Phase 2 does not increase the amount of secret information shared by Alice with each terminal (M_i), but it "redistributes" this information, such that all terminals share the same secret information. Said differently, if Eve knows nothing about the pair-wise secrets created at the end of phase 1, she will necessarily know nothing about the group secret created at the end of phase 2. However, as stated earlier, to create a perfect pair-wise secret with terminal T_i, Alice needs to know a lower bound for the number of x-packets shared with T_i Eve has missed, and she must set M_i (the size of the shared pair-wise secret) accordingly. The question remains: how can Alice estimate, in practice, a good lower bound for the number of x-packets shared with T_i that Eve has missed?

Avoiding the worst-case scenario. Our worst-case scenario is that Eve overhears all the x-packets received by a terminal T_i. In that case, Alice must create a shared pair-wise secret of size $M_i = 0$ with this terminal, which means that the shared group secret will also have size $L = min\{M_1, M_2, \ldots M_{n-1}\} = 0$.

To decrease the probability of this scenario, we make the terminals take turns in playing Alice's role (transmitting x-packets and constructing linear combinations). The idea is to make each terminal T_i receive information through multiple different channels (as opposed to receiving information only from Alice), making it unlikely that Eve is able to collect the same information as T_i. In a wireless network, Eve can collect the same information as terminal T_i only when, for *every* single terminal $T_{j \neq i}$, the channel between T_j and T_i happens to be the same with the channel between T_j and Eve throughout the protocol. This never happened in any of the experiments that we ran.

Linear combinations. The constructions we use to derive the y-packets, z-packets, and s-packets are based on Maximum Distance Separable (MDS) codes [10]. Due to space restrictions, we describe them in a separate report [9].

3.3 Lower-bounding What Eve Is Missing

The amount of information missed by Eve depends on channel conditions as well as Eve's network presence (the number and quality of antennas that she has at her disposal). The terminals may be able to influence and/or estimate channel conditions, but they have no way of knowing Eve's network presence. However, they can be more or less conservative when creating their secrets, depending on the strength of the adversary against whom they want to secure their communications.

One idea we are exploring is to artificially create channel conditions that are favorable to our protocol: In our setup, Bob and Eve are connected to the same broadcast network, so, when Alice transmits, it is possible that Eve misses none of the x-packets received by Bob (which means that Alice and Bob cannot create any shared secret at all). To prevent this scenario, we can use especially crafted interference that causes Eve to miss some minimum fraction of the packets shared by Alice and Bob, independently from the naturally occurring channel conditions. As a proof of concept, we are first trying out dedicated "interferer" nodes, but, ultimately, the terminals themselves could generate artificial

interference. iJam [8] uses a similar approach (but performs sophisticated OFDM-specific jamming, whereas we would only require the terminals to create enough noise to erase a minimum fraction of the packets from Eve's receiver).

Another idea we are exploring is to empirically estimate the amount of information missed by Eve based on the amount of information missed by the terminals. For instance, suppose the terminals want to be secure against an adversary that has the same kind of equipment as any terminal. We can pretend that each terminal T_j is Eve and that all the other $n-2$ terminals want to create pair-wise secrets with Alice that are unknown to T_j. Since we know which bits were received by each terminal (including T_j), we can compute exactly what the size of these supposed pair-wise secrets should be. Suppose we compute that, if terminal T_j was Eve, Alice and Bob should create a shared pair-wise secret of size M^j between them. We conservatively set the size of the shared pair-wise secret between Alice and Bob to $min\{M^2, \ldots M^{n-1}\}$. Similarly, to secure against an adversary that has as many antennas as k terminals, we can pretend that each set of k terminals *together* are Eve, and that all the other $n-k$ terminals want to create pair-wise secrets with Alice that are unknown to the k Eve-terminals.

4. DEPLOYMENT

We set up a small indoor wireless testbed that covers a square area of 14 m^2. We deployed $n = 8$ terminals and one adversary. All nodes are Asus WL-500gP wireless routers running 802.11g (at 2.472 GHz, transmit power 3 dBm) in ad-hoc mode, positioned within line of sight of each other. During our experiments, when a terminal transmits, it sends 100-byte packets at 1 Mbps.

We divide the testbed area in 9 logical cells, place Eve in one of them, and the terminals in various positions around her, but not in the same cell. Our rationale is the following: if a group of wireless nodes want to exchange a secret, it is reasonable to require from each of them to stand at least some minimum distance away from any other wireless node. In our testbed, this minimum distance is 1.75 m (the diagonal of a logical cell), and it was determined by the shape of the interferers' beams (a narrower beam would have led to a smaller minimum distance).

To generate interference, we use 6 WARP (Wireless Open-Access Research Platform) nodes, each with two directional antennas, each with a narrow 3-dB 22-degree beam. We place the interfering antennas along the perimeter of the covered area; we turn them on and off, such that, at any point in time, one pair of antennas creates noise along a row, while another pair creates noise along a column. The point of the artificial interference is to ensure that Eve misses some minimum fraction of the packets shared by Alice and Bob, independently from the naturally occurring channel conditions.

When we refer to an "experiment," we mean that we place n terminals and Eve on our testbed area, such that each cell is occupied by at most one node, and we run one round of our protocol. We run one such experiment for each possible positioning of n terminals and Eve, and we run one such set of experiments for $n = 3$ to 8 terminals. Each experiment is divided in time slots; at the beginning of each time slot, we turn on different interferers, such that, by the end of the experiment, we have rotated through all 9 noise patterns.

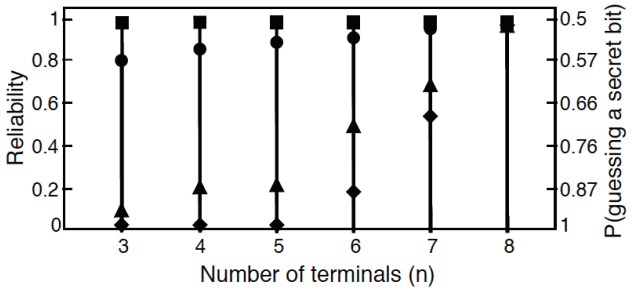

Figure 2: Reliability achieved by our protocol. Minimum (diamonds), 95th percentile (triangles), average (circles), and 50th percentile (squares).

We use two metrics to evaluate our protocol: The *efficiency* achieved by the protocol during an experiment is the number of shared secret bits generated by the terminals divided by the total number of bits that the terminals transmitted during the experiment. The *reliability* achieved by the protocol during an experiment represents the quality of the created shared secret; reliability r means that Eve can correctly guess each bit of the shared group secret with probability 2^{-r}.

Preliminary Results. For $n = 8$ terminals, we achieve minimum efficiency 0.038; given that the terminals transmit at rate 1 Mbps, this efficiency yields 38 secret Kbps.

Figure 2 shows the reliability achieved by our protocol as a function of the number of terminals n. For each value of n, we show the minimum (diamonds) and average (circles) reliability achieved across all experiments with n terminals, as well as the minimum reliability achieved during 95% (triangles) and 50% (squares) of the experiments with n terminals.

For $n = 8$ terminals, we achieve minimum reliability $r_{min} = 1$, i.e., Eve never learns anything about the secret. For $n = 6$ terminals, $r_{min} = 0.2$, i.e., Eve can correctly guess the value of a secret bit at most with probability $2^{-0.2} = 0.87$, but the value of an entire s-packet with probability $2^{-0.2 \cdot 800} \approx 0$ (each packet consists of 800 bits). Reliability decreases as the number of terminals decreases for the following reason: we estimate the length of the secret to create based on information provided by the terminals; the fewer the terminals, the less accurate the estimate, hence the more likely we are to create a longer secret than we should. However, for any number of terminals, in at least half of the node placements we achieve minimum reliability 1 (the 50th percentile is always 1).

These results tell us that, by using artificial interference, it is feasible to generate thousands of secret bits per second among a group of wireless nodes, such that an adversary

with a simple commodity receiver learns nothing about these secret bits. Of course, we would not do as well in the presence of a stronger adversary with multiple antennas. However, this result is good enough to give us hope that, even in the presence of a stronger adversary, we can achieve a non-zero secret bitrate.

5. RELATED WORK

Wyner introduced the wire-tap channel, where Alice transmits information to Bob, and a "wiretapper" Eve receives a noisy version of what Bob receives; he showed that Alice and Bob can achieve non-zero secrecy rate when the Alice/Bob channel is better than the Alice/Eve channel [2]. Maurer extended the wire-tap channel with public discussion; he provided lower and upper bounds for the secrecy rate from Alice to Bob and showed that it can be non-zero even when the Alice/Bob channel is worse than the Alice/Eve channel [3]. This work proved the theoretical feasibility of perfect pair-wise secrets (assuming Alice is connected to Eve through an idealized channel).

More recently, researchers have started to propose concrete, implementable protocols for creating pair-wise shared secrets in wireless networks. Some of them rely on the time-varying nature of wireless channels and the fact that Alice and Bob can measure the (time-varying) channel between them whereas Eve cannot [11, 12, 13]. They achieve secret generation rates up to a few tens of bps (in modified 802.11 or 802.15 environments). However, they rely on channel changes to extract secret bits, hence they are better-suited for mobile environments (in static environments, they can be vulnerable to eavesdropping) [14]. In another proposal, Alice and Bob create pair-wise shared secrets by combining and heuristically condensing the frames that are transmitted between them only once (the assumption being that these frames are less likely to have been heard by an eavesdropper than the rest) [4]. This protocol has been implemented in an 802.11 environment, but there has not been any evaluation or discussion of its efficiency or reliability yet. In iJam (developed in parallel with our work), 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 a pair-wise secret [8]. iJam achieves a secret-generation rate up to 18 Kbps (in a modified 802.11 environment).

Our protocol requires neither a mobile environment with quick channel changes nor custom physical-layer operations that are specific to OFDM (or any other transmission scheme). The key idea behind it—to the best of our knowledge not leveraged by any existing protocol—is that a conservative estimate of the number of bits missed by Eve is sufficient for creating shared secrets at Kbps rates. And, unlike any of the existing protocols (which are fundamentally tied to pair-wise secrets), it can gracefully handle multi-party secret generation.

6. CHALLENGES

We expect that our biggest challenge will be to characterize the robustness of our protocol against an adversary that possesses multiple antennas. Such an adversary not only overhears more information, but may also be able to cancel out from her received signal some of the artificial interference, provided the multipath channels between Alice–Eve and interferers–Eve satisfy certain "separability" conditions. We are currently studying the probability of these conditions occurring as a function of the number of Eve's antennas.

Even though we are using ideas from information theory, we are not aiming for information-theoretic security (perfect shared secrets). That would require perfect knowledge of the amount of information known to Eve, and, not surprisingly, we have found this to be practically infeasible. On the other hand, we do not believe that this is reason enough to discard the idea of leveraging noisy communication to create secrecy. Rather, we should view it as a starting point, an opportunity to depart from the assumption that an adversary has to have computational limitations.

7. 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(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 Trans. Info. Theory*, vol. 39, 1993.

[4] S. Xiao, W. Gong, and D. Towsley, "Secure Wireless Communication with Dynamic Secrets," in *Proceedings of the IEEE INFOCOM Conference*, 2010.

[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] "The On-demand Video Consumer", *Survey*, 2012, `http://www.youtube.com/yt/advertise/research.html`.

[7] L. Lai, H. El Gamal, and H. V. Poor, "The Wiretap Channel with Feedback: Encryption over the Channel," *IEEE Trans. Info. Theory*, vol. 54(11), pp. 5059-5067, 2008.

[8] S. Gollakota and D. Katabi, "Physical Layer Wireless Security Made Fast and Channel Independent," in *Proceedings of the IEEE INFOCOM Conference*, 2011.

[9] M. J. Siavoshani, U. Pulleti, E. Atsan, I. Safaka, C. Fragouli, K. Argyraki, and S. Diggavi. "Exchanging Secrets Without Using Cryptography," *Technical Report*, 2011, `http://arxiv.org/abs/1105.4991`.

[10] F. J. Macwilliams and N. J. A. Sloane, "The Theory of Error Correcting Codes," *North-Holland*, 2006.

[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 Trans. Information Forensics and Security*, vol. 5(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 IPSN Conference*, 2010.

[14] S. Jana, S. N. Premnath, M. Clark, S. Kasera, N. Patwari, and S. Krishnamurthy, "On the Effectiveness of Secret Key Extraction from Wireless Signal Strength in Real Environments," in *Proceedings of the ACM MOBICOM Conference*, 2009.