Appears in *Distributed Computing and Cryptography*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 2, ACM, 1991.

On the Structure of Secret Key Exchange Protocols

(Extended Abstract)

Mihir Bellare[*] Lenore Cowen[†] Shafi Goldwasser[‡]

MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139

May 1989

Abstract

A secret key exchange protocol, which enables two parties to establish a common and secret key over public channels, is a fundamental primitive for private communication. We show here that any secret key exchange protocol has strong structural properties. Our results imply that one-way functions are necessary for secret key exchange. As a corollary we obtain that one-way functions are also necessary for oblivious transfer.

Our results also imply that the existence of a secret key exchange protocol implies the existence of strong bit committment schemes, where the prover has probability 0 of cheating.

[*]Supported in part by NSF grant CCR-87-19689 and DARPA Contract N00014-89-J-1988.
[†]Supported in part by NSF grant CCR-86-57527 and DARPA Contract N00014-89-J-1988.
[‡]Supported in part by NSF grant CCR-86-57527 and DARPA Contract N00014-89-J-1988.

1 Introduction

Modern cryptography is fundamentally concerned with the problem of secure private communication. Suppose two parties, Alice and Bob, wish to communicate privately over a public channel (for instance, a telephone line with an eavesdropper). If Alice and Bob are able to meet, privately, beforehand, and agree on some common secret key, then it becomes easy for them to achieve such private communication. But Alice and Bob might not be able to first meet in private and agree on a key. In this case, we ask under what assumptions they can still agree on a common secret key, where their conversation is conducted entirely in public.

A Secret Key Exchange is a protocol where Alice and Bob, having no secret information in common to start, are able to agree on a common secret key, conversing over a public channel. The notion of a Secret Key Exchange protocol was first introduced in the seminal paper of Diffie and Hellman [DH]. [DH] presented a concrete implementation of a Secret Key Exchange protocol, dependent on a specific assumption (a variant on the discrete log), specially tailored to yield Secret Key Exchange. Secret Key Exchange is of course trivial if trapdoor permutations exist. However, there is no known implementation based on a weaker general assumption.

Recently Impaggliazo and Rudich [IR] showed that any "natural" proof that secret key exchange was possible with one way permutations would imply a proof that $P \neq NP$. We can look at the Impaggliazo and Rudich result as a negative one: it is hard to prove that the existence of a one-way permutation is sufficient to implement a Secret Key Exchange protocol.

In this paper we look at the other side of the coin. We ask, what can one say about the conditions necessary for Secret Key Exchange, and what is its power to yield other important primitives? We show that any Secret Key Exchange protocol has strong structural properties. We define the *Matrix of Conversations* for any Secret Key Exchange protocol and the results we obtain follow directly from our anaylsis of monochromatic blocks of this matrix. We show the following:

(1) The existence of a Secret Key Exchange protocol implies the existence of a one-way function. As a corollary, we immediately get that oblivious transfer implies one way functions. Coupled with results of [ILL] and [GGM], our results also imply that Secret Key Exchange is by itself enough to obtain pseudo-random bit generation, and private key encryption secure against chosen cyphertext attack. We obtain these theorems in both the uniform and non-uniform models, under two slightly different definitions of Secret Key Exchange. [IL] can independently obtain our result that Secret Key Exchange implies the existence of a one way function in the non-uniform model (Theorem 4.6 in our paper), by viewing Secret Key Exchange as an "Identification Scheme".

(2) The existence of a Secret Key Exchange protocol is enough to obtain Bit Commitment, in a setting where the player who commits to the bit need not be computationlly bounded and will have probability 0 of cheating. This is of separate interest, because we do not know that bit commitment follows from a one-way function with uniform security assumptions. This implies that we need only assume the existence of a Secret Key Exchange protocol to create envelopes, and hence carry out zero knowledge proofs.

2 Definitions

A Secret Key Exchange protocol involves two probabalistic polynomial time interactive Turing machines, Alice and Bob. Each machine has its own (private) worktape, and a private random tape (this last to model the fact that Alice and Bob can flip coins in private.) In addition there is a common, public, tape on which both Alice and Bob can read and write; Eve, a probabilistic polynomial time Turing machine eavesdropping on their conversation, can read this tape as well.

If our probabilistic polynomial time Turing machines are A and B, we let R_A and R_B denote their respective random tapes. For notational convenience, we assume that the protocol has the following form: in round i, A "speaks" first, by writing some string α_i on the shared conversation tape, and then B_i speaks. (Here, α_i is some function of R_A and the earlier portion of the conversation; A performs any necessary computation before she writes, and similarly for B). Then if we let $conv(R_A, R_B)$ denote the conversation between A and B when their random tapes are R_A and R_B, we have that $conv(R_A, R_B) = (\alpha_1, \beta_1, \ldots, \alpha_t, \beta_t)$ where

$$\begin{aligned} A(R_A, \alpha_1\beta_1 \ldots \alpha_{i-1}\beta_{i-1}) &= \alpha_i \\ B(R_B, \alpha_1\beta_1 \ldots \alpha_{i-1}\beta_{i-1}\alpha_i) &= \beta_i \ . \end{aligned}$$

There is a security parameter k, and for convenience we assume that $|R_A| = |R_B| = k$.

Each of A and B has a *key obtaining algorithm* (respectively denoted K_A and K_B). These are polynomial time functions which each party can compute on the conversation and his own coin tosses to yield his estimate of the key. For any instance (i.e. fixed choice of R_A, R_B) of the protocol we say that A and B have *agreed* on a key if

$$K_A(R_A, conv(R_A, R_B)) = K_B(R_B, conv(R_A, R_B)) \ ;$$

this common value is the key they have agreed on. The probability of agreement is thus

$$P_{agree} = \boldsymbol{P}(K_A(R_A, C) = K_B(R_B, C) : R_A, R_B \leftarrow \{0,1\}^k; C \leftarrow conv(R_A, R_B)) \ .$$

(For those unfamiliar with this notation, please see the appendix.) We are now ready to define Secret Key Exchange.

Definition 2.1 A protocol constitutes a *secret key exchange (SKE)* if there is an $\alpha > 0$ such that we have

- **Agreement:** $P_{agree} \geq \alpha$, and on each instance, A and B *know* whether or not they agreed
- **Secrecy:** For all probabilistic polynomial time algorithms E, for all $d > 0$ and all sufficiently large k,

$$\boldsymbol{P}(E(C) = K_A(R_A, C) = K_B(R_B, C) : R_A, R_B \leftarrow \{0,1\}^k; C \leftarrow conv(R_A, R_B)) \leq k^{-d} \ .$$

A few words of explanation and justification are needed for our requirement that A and B know whether or not they have agreed at the end of the SKE protocol. This requirement is left out of the definition of SKE given in [IR], which we will call *Weak Secret Key Exchange.*

Definition 2.2 A protocol constitutes a *Weak Secret Key Exchange (weak SKE)* if there is an $\alpha > 0$ such that we have

- **Agreement:** $P_{agree} \geq \alpha$
- **Secrecy:** For all probabilistic polynomial time algorithms E, for all $d > 0$ and all sufficiently large k,

$$\boldsymbol{P}(E(C) = K_A(R_A, C) = K_B(R_B, C) : R_A, R_B \leftarrow \{0,1\}^k; C \leftarrow conv(R_A, R_B)) \leq k^{-d} \ .$$

Weak SKE though, is quickly seen to be too weak to capture any real notion of Secret Key Exchange (We remark that since [IR] prove that it is hard to prove the existence of even a Weak SKE protocol from one-way permutations, using this weaker definition only strengthens their results). To see the problems with this definition, look at what happens in a weak SKE protocol when A and B disagree. They may not even know that they have done so and, attempting to communicate based on different keys, be unable to understand each other. One way to avoid this is through redundancy: perform many key exchanges and then use all keys to send duplicate copies of messages simultaneously. But although the communication will now succeed with high probability because there will probably be a common key amongst the many keys used, the eavesdropper might be able to understand the messages based on keys other than the one agreed on. A related problem is that one usually wants, from the key, to extract a single hard bit since the eavesdropper might, for example, know a few bits of the key; but in this case even the redundancy approach will fail to supply a common key.

One might imagine from this discussion that a much stronger notion of key exchange is required to attain anything reasonable. However, the condition we added to weak SKE: that A and B know whether or not they agree, will suffice, since A and B only take action (communication, extracting a hard bit) when they do agree. Since this is after all what "agreement" means, this is only the natural definition.

Nevertheless, we show that even just assuming the existence of a weak SKE protocol implies one-way function exist, but for this weak case, we need non-uniform security assumptions. In fact, we present most of the structural results in section three in terms of weak SKE, since the existence of a weak SKE protocol is already enough to guarantee the strong structural properties that we are interested in.

Finally, in our definition of (strong) SKE the following theorem shows that just assuming that A and B know when they agree means that an an even stronger property holds. Namely, we may assume the ideal case, that A and B *always* agree.

Theorem 2.3 Suppose we have a SKE protocol. Then there is another SKE protocol in which A and B agree with probability 1.

Proof: A and B can execute the original SKE

$$\frac{k}{\lg \frac{1}{1-\alpha}}$$

times. If any of these runs yields agreement, they take as their key the one from the first agreeing run. If all runs yield disagreement they agree on some default value (say 0^k) as their key. Eve can certainly find the key in the latter case, but the probability that this case occurs is $\leq 2^{-k}$. On the other hand, the agreement probability is 1. ■

These definitions are in the uniform setting; in the corresponding non-uniform setting Alice, Bob, and Eve would all be polynomial size circuits.

3 Structure of SKE

We show here that certain structural properties are inherent in a secret key exchange protocol.

Fix any weak SKE protocol. Since any SKE protocol is also automatically a weak SKE protocol, our lemmas in this section apply to both SKE and weak SKE protocols alike.

3.1 Terminology

The *matrix of conversations* of the SKE is the 2^k by 2^k matrix whose row i column j entry is $conv(i, j)$ (here $0 \leq i, j < 2^k$ are identified with the corresponding binary strings of length k). This represents a convenient way to summarize the information as to the conversations taking place.

We say that $R_A R_B$ is an *agreement point* if $K_A(R_A, conv(R_A, R_B)) = K_B(R_B, conv(R_A, R_B))$; otherwise it is a *disagreement point*. It is an agreement/disagreement point of C if $conv(R_A, R_B) = C$. If $R_A R_B$ is an agreement point we denote the agreed on key by $K(R_A, R_B)$.

3.2 Basic "Rectangle" Lemmas

The following "rectangle" property is a well known fact about protocols:

Lemma 3.1 Suppose $R_A^1, R_A^2, R_B^1, R_B^2$ are such that $conv(R_A^1, R_B^1) = conv(R_A^2, R_B^2)$. Then in fact $conv(R_A^1, R_B^1) = conv(R_A^1, R_B^2) = conv(R_A^2, R_B^2) = conv(R_A^2, R_B^1)$.

This enables us to show that the set of entries in the matrix of conversations which are equal to a fixed conversation C form a submatrix.

Lemma 3.2 Let C be a conversation, and let

$$\mathcal{R}(C) = \{(R_A, R_B) : conv(R_A, R_B) = C\} .$$

Then there exist sets $\mathcal{R}_A(C), \mathcal{R}_B(C) \subseteq \{0,1\}^k$ such that $\mathcal{R}(C) = \mathcal{R}_A(C) \times \mathcal{R}_B(C)$.

Proof: Let

$$\begin{aligned}
\mathcal{R}_A(C) &= \{R_A : \text{ there exists } R_B \text{ such that } conv(R_A^1, R_B^1) = C\} \\
\mathcal{R}_B(C) &= \{R_B : \text{ there exists } R_A \text{ such that } conv(R_A^2, R_B^2) = C\} .
\end{aligned}$$

Clearly $\mathcal{R}(C) \subseteq \mathcal{R}_A(C) \times \mathcal{R}_B(C)$. So it suffices to show that $\mathcal{R}_A(C) \times \mathcal{R}_B(C) \subseteq \mathcal{R}(C)$. So suppose $R_A^1 \in \mathcal{R}_A(C)$ and $R_B^2 \in \mathcal{R}_B(C)$. Thus there exists R_B^1 such that $conv(R_A^1, R_B^1) = C$ and R_A^2 such that $conv(R_A^2, R_B^2) = C$. Hence by Lemma 3.1 we get $conv(R_A^1, R_B^2) = C$. Thus $(R_A^1, R_B^2) \in \mathcal{R}(C)$. ■

We now proceed to examine in more detail this submatrix of points with the same conversation.

3.3 Agreement Matrix of a Conversation

The *agreement matrix* of the conversation C, denoted by $M(C)$, is an m by n matrix where $m = |\mathcal{R}_A(C)|$ and $n = |\mathcal{R}_B(C)|$. The rows represent elements of $\mathcal{R}_A(C)$ while the columns represent elements of $\mathcal{R}_B(C)$. The row R_A column R_B entry of $M(C)$ is $K(R_A, R_B)$ in the case that $K_A(R_A, C) = K_B(R_B, C)$ and the special symbol $*$ otherwise; that is, this entry is the agreed on key in the case that agreement is reached, and $*$ in the case of disagreement. The agreement matrix of C captures all the information about what happens when the conversation is C.

We examine the structure of $M(C)$ in the case that C has lots of agreement points. We begin with a lemma about points of $M(C)$ where key agreement is reached, but the agreement is on *different* keys. We can conclude that this implies the existence of disagreement points for this conversation.

Lemma 3.3 Suppose $R_A^1, R_A^2, R_B^1, R_B^2$ are such that

- $conv(R_A^1, R_B^1) = conv(R_A^2, R_B^2) = C$
- $K_A(R_A^1, C) = K_B(R_B^1, C) = \kappa_1$
- $K_A(R_A^2, C) = K_B(R_B^2, C) = \kappa_2$
- $\kappa_1 \neq \kappa_2$.

Then $R_A^1 R_B^2$ and $R_A^2 R_B^1$ are disagreement points of C.

Proof: Suppose A has R_A^1 and B has R_B^2. By Lemma 3.1 we have $C = conv(R_A^1, R_B^2)$. So A will get $\kappa_1 = K_A(R_A^1, C)$ and B will get $\kappa_2 = K_B(R_B^2, C)$. Similarly if A has R_A^2 and B has R_B^1 then A will get κ_2 and B will get κ_1. The lemma follows from the fact that $\kappa_1 \neq \kappa_2$. ■

This enables us to show that if C has no disagreement points then the structure of this matrix is exceedingly simple: all its entries are the same.

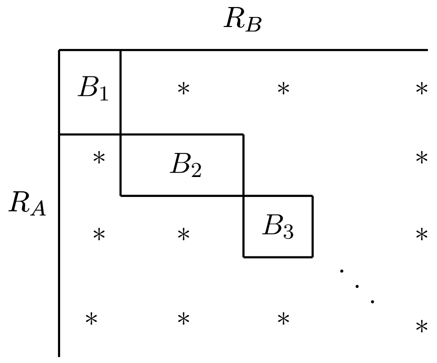


Figure 1: $M(C)$ in normal form: each block B_i is the set of all points at which agreement is on a particular key κ_i.

Lemma 3.4 Suppose C has no disagreement points. Then there exists a single key κ such that the key agreed on when the conversation is C is always κ.

Proof: If there were $R_A^1, R_B^1, R_A^2, R_B^2$ such that

$$K_A(R_A^1, C) = K_B(R_B^1, C) \neq K_B(R_B^2, C) = K_A(R_A^2, C) ,$$

then (by Lemma 3.3) C would have disagreement points: namely $R_A^1 R_B^2$ and $R_A^2 R_B^1$. ■

Another useful fact to note is

Lemma 3.5 Let κ be a key. The set of entries of $M(C)$ equal to κ forms a submatrix of $M(C)$.

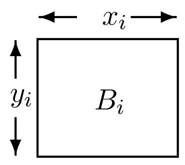
Proof: Clear ■

Even if there are some disagreement points, we can show that a fair number of agreement points still do have the same key:

Lemma 3.6 Suppose that the number of agreement points in $M(C)$ is $\geq \frac{mn\alpha}{2}$. Then there exists a single key κ such that $\geq \frac{mn\alpha^2}{16}$ entries of $M(C)$ are equal to κ.

Proof: We first put $M(C)$ into a normal form. We pick a key κ_1 which occurs in the matrix, and, by permuting rows and columns, move the submatrix (Lemma 3.5) of entries equal to this key to the upper left corner to get a block we call B_1. We are then assured that points directly to the right of this block as well as points directly below this block are all $*$. We then take another key κ_2 and move it as far up and left as possible in the remaining matrix (the one with the rows and columns of B_1 removed). Continuing in this way we obtain the structure of Figure 1.

We let x_i and y_i denote the length and breadth respectively of block B_i; that is:



We wish to show that there is an i with $x_i y_i \geq \frac{mn\alpha^2}{16}$. We visualize this geometrically: we have a set of rectangles in the plane, laid out in the particular pattern of Figure 1, and we know that they cover a total area of $\frac{mn\alpha}{2}$. We have to show that there is a single rectangle of large enough area.

We claim that we can assume, without loss of generality, that $m = n$. For if this is not the case then we can apply a linear transformation of the plane (say $(x, y) \mapsto (\frac{\sqrt{mn}}{n}x, \frac{\sqrt{mn}}{m}y))$ via which the m by n rectangle is turned into a $\sqrt{mn}$ by $\sqrt{mn}$ square, and then use the fact that linear transformations preserve ratios of areas.

Now let $N = m = n$. Assume towards a contradiction that $x_i y_i < \frac{N^2\alpha^2}{16}$ for all i. Let $M_i = \max(x_i, y_i)$ and $m_i = \min(x_i, y_i)$. Then $m_i < \sqrt{\frac{N^2\alpha^2}{16}}$ for all i. So

$$\begin{aligned}
\sum_i x_i y_i &= \sum_i M_i m_i \\
&< \sqrt{\frac{N^2\alpha^2}{16}} \sum_i M_i \\
&\leq \sqrt{\frac{N^2\alpha^2}{16}} (2N) \\
&= \frac{N^2\alpha}{2} ,
\end{aligned}$$

contradicting the fact that by assumption

$$\sum_i x_i y_i \geq \frac{N^2\alpha}{2} .$$

■

3.4 Distribution of "Good" Conversations

Since, by definition of a (weak) SKE protocol, agreement occurs with $\geq \alpha$ probability over all possible runs of the protocol, there must be a set of conversations on which, when A and B have one of these conversations, then it is fairly likely they will reach agreement. We call these conversations "good".

Definition 3.7 A conversation C is *good* if $M(C)$ has $\geq \frac{mn\alpha}{2}$ agreement points.

Lemma 3.8 The probability of a good conversation is $\geq \frac{\alpha}{2}$.

Proof: Apply the pigeonhole principle. ■

4 Applications

4.1 One-Way Functions

Definition 4.1 A polynomial time computable function $f = \{f_k\}$ (where $f_k : \{0,1\}^k \to \{0,1\}^k$) is *one way* if for all probabilistic polynomial time algorithms I, for all $c > 0$ and sufficiently large k,

$$\boldsymbol{P}(f(y) = f(x) : x \leftarrow \{0,1\}^k; y \leftarrow I(f(x))) < k^{-c} .$$

This is the uniform definition; for the non-uniform one, change everything to poly-sized circuits.

The concept of an informationally one-way function was introduced in [ILL]. We give only an informal definition here:

Definition 4.2 A polynomial time computable function $f = \{f_k\}$ is informationally one-way if there is no probabilistic polynomial time algorithm which (with probability of the form $1 - k^{-e}$ for some $e > 0$) returns on input $y \in \{0,1\}^k$ a random element of $f^{-1}(y)$.

In the non-uniform setting [ILL] show that these are not weaker than one-way functions:

Theorem 4.3 [ILL] (non-uniform) The existence of informationally one-way functions implies the existence of one-way functions.

We will stick to the convention introduced above of saying "non-uniform" before the theorem statement when the theorem makes use of non-uniformity. It should be understood that if nothing is said then the result holds for both the uniform and the non-uniform models.

We now have

Theorem 4.4 SKE implies the existence of a one-way function.

Proof: By Theorem 2.3 we may assume that we have an SKE protocol in which A and B always agree. We now show that the function $f(R_A R_B) = conv(R_A, R_B)$ is one way. Assume not. We thus have an inverting algorithm I for f. We use this to construct an eavesdropper E who breaks the secrecy of the SKE. E works exactly as expected: on input a random conversation C she evaluates $I(C) = R_A R_B$ and then outputs $K(R_A, R_B)$ as her estimate of the key. Lemma 3.4 (which applies since the agreement probability is 1) now provides the crucial step: it says that this is indeed the key that A and B agreed on. The eavesdropper is thus succeeding with exactly the same probability that I is inverting f, contradicting the secrecy of the SKE. ∎

If we allow non-uniformity we can prove that even weak SKE implies the existence of one-way functions. First, via Lemma 3.6, we get informationally one-way functions:

Theorem 4.5 Weak SKE implies the existence of an informationally one-way function.

Proof: The idea of the proof is to show that the function $f(R_A R_B) = conv(R_A, R_B)$ is informationally one-way. Suppose not. Let I be an algorithm which on input C returns a random element of $f^{-1}(C)$ with probability $\geq 1 - k^{-e}$. This algorithm is thus sampling the agreement matrix $M(C)$. Since good conversations have a fairly large probability (Lemma 3.8) we hit them fairly often. Whenever we do, Lemma 3.6 says that we break the key exchange with probability $\geq \frac{\alpha^2}{16}$. Putting all this together, we get a contradiction to the secrecy of the key exchange. ∎

It now follows from Theorem 4.3 that

Theorem 4.6 (non-uniform) Weak SKE implies the existence of a one-way function.

4.2 Bit Committment

We show that SKE implies strong committer bit committment (bit committment where the committer cannot cheat on the decommittals even if he is not computationally bounded).

A *bit commitment* protocol involves two parties: D (the committer) and R (the receiver). The protocol has two stages

- **Stage 1 (Commitment):** . D has a bit b which she wishes to commit. She and the receiver exchange messages. At the end of the conversation R has some information that represents the bit b.
- Stage 2. (Decommitment) D gives R some additional information that reveals b to R.

The protocol must obey the following, for all $c > 0$ and large enough security parameter k:

- After the commit stage, R cannot guess b with probability greater than $\frac{1}{2} + k^{-c}$.
- In the decommitment stage, D can only reveal one value. If she tries to reveal a different value she is caught with probability at least $1 - k^{-c}$.

We say that the protocol is a *strong commiter* bit commitment if the property that the commiter can reveal only one value holds regardless of the computational power of the commiter.

In the bit commitment protocol we will now construct, the committer, D, will simulate a key exchange between A and B, and in some sense R, the receiver, will be in the position of the eavesdropper. First, however, we need the following generalization of the theorem of Goldreich and Levin [GL]:

Theorem 4.7 Suppose $f = \{f_k\}, g = \{g_k\}$ are polynomial time computable functions, where $f_k, g_k : \{0,1\}^k \to \{0,1\}^k$. Suppose that for all PPT algorithms I, all $c > 0$ and sufficiently large k,

$$\boldsymbol{P}(y = g(x) : x \leftarrow \{0,1\}^k \, ; \, y \leftarrow I(f(x))) < k^{-c} \, .$$

Then for all PPT algorithms I, all $c > 0$ and sufficiently large k,

$$\boldsymbol{P}(b = \mathcal{B}(g(x), r) : x, r \leftarrow \{0,1\}^k \, ; \, b \leftarrow I(f(x), r)) < \frac{1}{2} + k^{-c} \, ,$$

where $\mathcal{B}(g(x), r)$ is the inner product mod 2 of $g(x)$ and r. **Proof:** The proof of [GL] extends ■

Theorem 4.8 SKE implies (strong commiter) bit commitment.

Proof: By Theorem 2.3 we may assume that the agreement probability of the SKE protocol is 1. If the committer D wishes to commit to bit b she picks $x = R_A R_B$ at random and computes

$$\begin{array}{rclcl} f(x) & = & conv(R_A, R_B) & = & C \\ g(x) & = & K(R_A, R_B) & = & \kappa \, . \end{array}$$

She then picks r at random and sends $(\mathcal{B}(\kappa, r) \oplus b, C, r)$ to the receiver R. To decommit she reveals x.

The fact that this is a SKE protocol implies that $g(x)$ is hard to compute in probabilistic polynomial time given $f(x)$. Theorem 4.7 thus implies that R has no advantage in predicting b as long as he is PPT.

On the other hand, κ is uniquely determined given C (Lemma 3.4) so the probability that D can cheat and later reveal $\bar{b}$ is 0.

Naor [N] has shown that pseudo-random generators can be used for bit committment, and hence if we have non-uniform security assumptions one-way functions suffice [ILL]. The above is stronger in that

- It does not require non-uniformity
- The probability of cheating on the part of the committer is zero rather than just exponentially small as in Naor's scheme.
- The scheme is *non-interactive*; that is, the receiver does not have to talk in the commitment stage.

4.3 Other Implications

Clearly, anything that yields secret key exchange must imply the existence of one-way functions. Of particular interest is oblivious transfer:

Corollary 4.9 Oblivious transfer implies the existence of a one-way function.

Proof: Oblivious transfer implies SKE (Note: Earlier reductions may not have had the property that the parties know whether or not they have agreed, but we can get that). ∎

On the other hand the fact the secret key exchange yields one-way functions and bit committent implies that various much more powerful cryptographic primitives are possible given the existence of secret key exchange alone. Combining with results of [ILL] and [GGM] we get

Corollary 4.10 (non-uniform) (Weak) SKE implies pseudo random bit generators and private key encryption secure against chosen message attack.

Bit committment suffices for zero knowledge proofs [GMW] so

Corollary 4.11 SKE implies the ability to give zero knowledge proofs for any language in NP.

4.4 Agreement in SKE

We can show a 0-1 law for agreement in any SKE protocol:

Theorem 4.12 For any conversation in an SKE protocol, the probability of agreement on this conversation is either 0 or 1.

Proof: If there is one agreement point then the fact that A and B know whether or not they agree means that the row and column of this point also consist of agreement points. Applying the argument to these points, the whole matrix consists of agreement points. ∎

4.5 Perfect Secret Key Exchange

Secret Key exchange can also be defined in the "information theoretic" sense. That is, we allow A, B and E to be infinitely powerful. We call this perfect SKE. Recently Yung [Y] showed that perfect oblivious transfer is impossible. The same holds for secret key exchange, and this has been known; one proof is due to Killian [K] . We note that we can prove the impossibility of perfect SKE directly from our definition of the Matrix of Conversations.

Theorem 4.13 Perfect (weak) SKE is impossible.

Proof: E knows the agreement matrix of the conversation. For SKE this means, by Lemma 3.4, that she gets the key with the same probability that A and B agree on it. For weak SKE we use Lemma 3.6; E gets the key with probability $\frac{\alpha^2}{16}$. ∎

Acknowledgments

The first author thanks John Rompel and Su-Ming Wu for helpful discussions.

References

[DH] Diffie, W. and E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, **22**(5), 644-654 (November 1976).

[GGM] Goldreich, O., S. Goldwasser, and S. Micali, "How To Construct Random Functions," *Journal of the Association for Computing Machinery* **33**(4), 792-807 (October 1986).

[GL] Goldreich, O., and L. Levin "A Hardcore Predicate for Any One-Way Function," *Proceedings of the 21st Annual Symposium on the Theory of Computing*, ACM, 1989.

[GMR] Goldwasser, S., S. Micali, and R. Rivest, "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," *SIAM Journal on Computing* **17**(2), 281-308 (April 1988).

[GMW] Goldreich, O., S. Micali, and A. Wigderson, "Proofs that Yield Nothing but their Validity", *Proceedings of the 27th Symposium on Foundations of Computer Science*, IEEE, 1986.

[IL] Impagliazzo, R., and Luby, M., private communication.

[ILL] Levin, L., M. Luby and R. Impagliazzo, "Pseudo-Random Generation from One-Way Functions," *Proceedings of the 21st Annual Symposium on the Theory of Computing*, ACM, 1989.

[IR] Impagliazzo, R. and S. Rudich, "Limits on the Provable Consequences of One-Way Permutations," *Proceedings of the 21st Annual Symposium on the Theory of Computing*, ACM, 1989.

[K] Kilian, J., private communication.

[N] Naor, M., "Bit Committment Using Pseudo Randomness," *Advances in Cryptology – CRYPTO '89*, Lecture Notes in Computer Science Vol. 435, G. Brassard ed., Springer-Verlag, 1989.

[Y] Yung, M., "On the Impossibility of Perfect and One-Way Oblivious Transfer," manuscript.

A Appendix: Notation and Conventions

We take these notations and conventions for probabilistic algorithms from [GMR].

We emphasize the number of inputs received by an algorithm as follows. If algorithm A receives only one input we write "$A(\cdot)$"; if it receives two we write "$A(\cdot,\cdot)$", and so on.

If A is a probabilistic algorithm then, for any input i the notation $A(i)$ refers to the probability space which to the string σ assigns the probability that A, on input i, outputs σ. We point out the special case in which A takes no inputs; in this case A refers to the algorithm itself whereas the notation $A()$ refers to the probability space obtained by running A on no input.

If S is a probability space we denote by $\boldsymbol{P}_S(e)$ the probability that S associates with element e. We denote by $[S]$ the set of elements to which S assigns positive probability.

If $f(\cdot)$ and $g(\cdot,\cdots)$ are probabilistic algorithms then $f(g(\cdot,\cdots))$ is the probabilistic algorithm obtained by composing f and g (i.e. running f on g's output). For any inputs $x, y, \ldots$ the associated probability space is denoted $f(g(x,y,\cdots))$.

If S is a probability space then $x \leftarrow S$ denotes the algorithm which assigns to x an element randomly selected according to S (that is, x is assigned the value e with probability $\boldsymbol{P}_S(e)$). If S is a finite set, $x \leftarrow S$ denotes the operation of selecting an element of S uniformly at random (in the case that the set is of only one element e we write $x \leftarrow e$ rather than $x \leftarrow \{e\}$).

For probability spaces $S, T, \ldots$, the notation

$$\boldsymbol{P}(p(x,y,\cdots) : x \leftarrow S; y \leftarrow T; \cdots)$$

denotes the probability that the predicate $p(x, y, \cdots)$ is true after the (ordered) execution of the algorithms $x \leftarrow S$, $y \leftarrow T$, etc. The notation

$$\{ f(x, y, \cdots) : x \leftarrow S; y \leftarrow T; \cdots \}$$

denotes the probability space which to the string σ assigns the probability

$$\boldsymbol{P}(\sigma = f(x, y, \cdots) : x \leftarrow S; y \leftarrow T; \cdots) ,$$

f being some function.

We let PPT denote the set of probabilistic (expected) polynomial time algorithms.