Pseudorandom Function Tribe Ensembles Based on One-Way Permutations: Improvements and Applications

Marc Fischlin

Fachbereich Mathematik (AG 7.2)
Johann Wolfgang Goethe-Universität Frankfurt am Main
Postfach 111932
60054 Frankfurt/Main, Germany
marc@mi.informatik.uni-frankfurt.de
http://www.mi.informatik.uni-frankfurt.de/

June 29, 1999

Abstract. Pseudorandom function tribe ensembles are pseudorandom function ensembles that have an additional collision resistance property: almost all functions have disjoint ranges. We present an alternative to the construction of pseudorandom function tribe ensembles based on one-way permutations given by Canetti, Micciancio and Reingold [CMR98]. Our approach yields two different but related solutions: One construction is somewhat theoretic, but conceptually simple and therefore gives an easier proof that one-way permutations suffice to construct pseudorandom function tribe ensembles. The other, slightly more complicated solution provides a practical construction; it starts with an arbitrary pseudorandom function ensemble and assimilates the one-way permutation to this ensemble. Therefore, the second solution inherits important characteristics of the underlying pseudorandom function ensemble: it is almost as efficient and if the starting pseudorandom function ensemble is efficiently invertible (given the secret key) then so is the derived tribe ensemble. We also show that the latter solution yields so-called committing private-key encryption schemes. i.e., where each ciphertext corresponds to exactly one plaintext — independently of the choice of the secret key or the random bits used in the encryption process.

1 Introduction

In [CMR98] Canetti, Micciancio and Reingold introduce the concept of pseudorandom function tribe ensembles. Informally, such tribe ensembles consist of pseudorandom functions that have an independent public key in addition to the secret key. Though this public key, called the tribe key, is independent of the secret key, it guarantees that any image/preimage pair commits to the secret key. More specifically, for a random tribe key t there do not exist secret keys $k \neq k'$ and a value x such that the functions determined by the keys k, t

resp. k', t map x to the same value (except with exponentially small probability, where the probability is taken over the choice of t). Canetti et al. [CMR98] use such pseudorandom function tribe ensembles to construct perfectly one-way probabilistic hash functions. In contrast to ordinary one-way functions, such perfectly one-way probabilistic hash functions hide all partial information about the preimage (secrecy), yet finding a hash value together with distinct preimages is infeasible (collision resistance). In [C97] Canetti presents perfectly one-way hash functions based on a specific number-theoretic assumption, namely the Decisional-Diffie-Hellman assumption. Generalizing this result, Canetti, Micciancio and Reingold [CMR98] show that perfectly one-way functions can be constructed from any cryptographic hash function (achieving secrecy statistically and collision resistance computationally) or from any pseudorandom function tribe ensembles (with computational secrecy and statistical collision resistance). In the latter case, the pseudorandomness of the tribe ensemble provides secrecy and collision resistance follows from the property of the tribe key. Canetti et al. [CMR98] also prove that PRF tribe ensembles exist if one-way permutations exist. Their construction is a modification of the GGM-tree design of PRF ensembles [GGM86] combined with a generalization of the Goldreich-Levin hardcore predicate [GL89]. A sketch of this construction is given in Appendix A. Here, we take a different approach which consists of two elementary and independent steps. First, we show that any one-way permutation suffices to construct a PRF ensemble such that for distinct secret keys k, k' the functions determined by k and k' map 1^n to different values. We call such ensembles fixed-value-key-binding as the key is determined by the function value for 1^n or, using a minor modification, for any other fixed value instead of 1^n. Second, we prove that fixed-value-key-binding PRF ensembles yield PRF tribe enembles. After presenting a conceptually simple construction of fixed-value-key-binding ensembles based on the GGM-tree design to the authors of [CMR98], they pointed out an improvement that led to the more practical solution which does not necessarily involve the GGM-construction. Instead it works with a every PRF ensemble by assimilating the one-way permutation to the given ensemble. This yields a fixed-value-key-binding PRF ensemble and, in turn, a PRF tribe ensemble which is almost as efficient as the starting PRF ensemble. Moreover, if the functions of the ordinary ensemble are efficiently invertible with help of the secret key then so are the functions of the tribe ensemble. From a theoretical and practical point of view this gives us the best of both worlds: As for the theory, we obtain a simple proof that the existence of one-way permutations implies the existence of PRF tribe ensembles. For practical purposes, we present a construction where pseudorandomness is slightly harder to prove, but which has nice properties. In both cases, the second step deriving the tribe ensemble from the fixed-value-key-binding ensemble is identical. We give an outline of this part. It is a reminiscent of Naor's statistically-binding bit commitment scheme [N91]. There, the receiver sends a random $3n$-bit string A to the committing party who applies a pseudorandom generator $G : \{0,1\}^n \to \{0,1\}^{3n}$ to a random value $r \in \{0,1\}^n$ and returns $G(r) \oplus A$ to commit to 1 resp. $G(r)$ to commit to 0. The receiver cannot distinguish both cases with significant advantage because of the pseudorandomness of the generator's output. On the other hand, to open a commitment ambiguously the sender has to find r, r' such that $G(r) = G(r') \oplus A$. But $\#\{G(r) \oplus G(r') \mid r, r'\} \leq 2^{2n}$, hence $A \in \{G(r) \oplus G(r') \mid r, r'\}$ with probability at most 2^{-n} (over the choice of A). This means that the commitment cannot be opened ambiguously with probability at least $1 - 2^{-n}$. We adopt this idea to define our PRF tribe ensemble. Given a fixed-value-key-binding PRF ensemble we define an appropriate fixed-value-key-binding PRF ensemble F^{stretch} with functions f_k^{stretch} that stretch the input to a sufficiently large output. We then show that there exists a value I_k (depending on the secret key k) and a function $\mathrm{XOR}(t, I_k)$ of the tribe key t and I_k such that from the key-binding property it follows that for different keys k, k' and random t the value $\mathrm{XOR}(t, I_k) \oplus \mathrm{XOR}(t, I_{k'})$ is a uniformly distributed string having the same length as the output of f_k^{stretch}.[1] In other words, $\mathrm{XOR}(t, I_k)$ is an xor universal hash function [CW79] with argument I_k and description t. Define the functions f_k^t of the PRF tribe ensemble by $f_k^t(x) = f_k^{\mathsf{stretch}}(x) \oplus \mathrm{XOR}(t, I_k)$. A collision $f_k^t(x) = f_{k'}^t(x)$ for $x, k \neq k'$ implies

$$f_k^{\mathsf{stretch}}(x) \oplus f_{k'}^{\mathsf{stretch}}(x) = \mathrm{XOR}(t, I_k) \oplus \mathrm{XOR}(t, I_{k'})$$

Since the output length of the functions in F^{stretch} is much bigger than the input length and as $\mathrm{XOR}(t, I_k) \oplus \mathrm{XOR}(t, I_{k'})$ is a random string for random t, collision resistance of the tribe ensemble is obtained as in Naor's bit commitment scheme. Additionally, we will show that the pseudorandomness of the tribe ensemble follows from the pseudorandomness of F^{stretch}.

Finally, based on our PRF tribe ensemble, we present a committing private-key encryption scheme, i.e., such that one cannot later open an encryption ambiguously by pretending to have used a different secret key. Secure committing *public*-key encryption systems can be derived for example from trapdoor permutations using the Goldreich-Levin hardcore predicate. In fact, constructing the opposite, public-key schemes that allow to open encryptions ambiguously, is a very interesting problem, because such schemes yield multiparty protocols secure against adaptive adversaries [CFGN96,CG96,CDNO97]. Given an arbitrary fixed-value-key-binding PRF ensemble we present a straightforward solution for a committing private-key system. Unfortunately, this scheme allows to deduce if two encryptions have been generated with the same secret key; a drawback which schemes based on PRF ensembles usually do not have. Therefore, we present another committing system that does not have this disadvantage, and prove that this scheme is secure against chosen ciphertext and plaintext attacks or, equivalent, non-malleable.

Our committing private-key scheme has an interesting application in light of the recent announcement of the French government [CISI99]. The government plans a liberalisation of encryption and, in particular, it is not planned to enforce key deposit. Instead, there are supposed to be obligations to hand over the

[1] Actually, this string will be uniformly distributed in a sufficiently large subset of the binary strings of the output length.

cleartext to legal authorities on request. For ordinary[2] public-key schemes, this means that the user simply hands over the message and the randomness and the authority checks the validity by re-encrypting and comparing the result to the ciphertext. As long as the encryption scheme supports errorless decryption, the user cannot claim to have encrypted a different message. Unfortunately, this does not work for private-key schemes in general, because the authority cannot re-encrypt without knowing the secret key, and even if it is given *some* secret key, this might not be the key that has been really used. Hence, the user might be able to open the encryption ambiguously. This is ruled out if we use a committing private-key encryption scheme: If a legal authority compels to open an encryption then the user hands over the message m, the randomness r and the secret key k. The authority is then able to verify that the encryption is opened correctly. But this also reveals the secret key and therefore all messages encrypted with this key. Zero-Knowledge proofs provide a solution that allows the user to maintain secrecy of the key: The user passes m and r to the authority and then proves in zero-knowledge that there exists some key k such that the ciphertext equals the encryption of m under key k with randomness r. Obviously, this is an NP-statement and can therefore be proven in constant-round zero-knowledge (either based on general proof systems or on some specific system for the underlying encryption scheme). Since there is only a single triple (m, r, k) that maps to the ciphertext in question, the soundness of the zero-knowledge proof guarantees that a dishonest user cannot open the encryption unambiguously except with small error probability. This method keeps k secret at a very high price: The computational effort is quite immense and it requires interaction between the authority and the user. The proof is, however, only necessary in case of a dispute and the obligations require interaction anyway. Additionally, if the authority obliges many messages encrypted with the same key then we prove the validity of the encryptions in parallel, i.e., prove that there is a key such that the sequence of ciphertexts have been produced from the sequence of messages.

2 Preliminaries

For sake of self-containment, we briefly recall basic definitions of pseudorandom functions, pseudorandom generators, etc. See [G95] for the underlying intuition. At the end of this section, we repeat the GGM-construction and the definition of pseudorandom function tribe ensembles. We present all definitions for uniform adversaries only; replacing the term "polynomial-time algorithm" by "polynomial circuit family" one easily obtains the nonuniform counterpart.

A function $\delta(n)$ is called *negligible in n* if $\delta(n) < 1/p(n)$ for any positive polynomial $p(n)$ and all sufficiently large n. A polynomial-time computable function f is *one-way* if for any probabilistic polynomial-time algorithm A the probability $\mathrm{Prob}\left[\, A(1^n, f(x)) \in f^{-1}(x) \right]$ that A outputs a preimage of $f(x)$ for random $x \in \{0,1\}^n$ is negligible in n. A one-way function f is a *one-way permutation*

[2] We do not deal with deniable schemes [CDNO97], but have practical systems like RSA- or ElGamal-based ones in mind.

if f permutes $\{0,1\}^n$ for every n. A *hardcore predicate* of a one-way function f is a polynomial-time computable predicate B such that for any probabilistic polynomial-time algorithm A it holds that $\text{Prob}[A(1^n, f(x)) = B(x)]$ for random $x \in \{0,1\}^n$ is negligible in n. According to a result of Goldreich and Levin [GL89] every one-way function can be modified to have a hardcore predicate. A polynomial-time computable function G is a *pseudorandom generator* if there exists some function $\ell(n)$ such that $\ell(n) > n$ and $G(x) \in \{0,1\}^{\ell(n)}$ for all $x \in \{0,1\}^n$ and all n, and such that for any probabilistic polynomial-time algorithm D the advantage $|\text{Prob}[D(G(x)) = 1] - \text{Prob}[D(y) = 1]|$ is neglible in n, where x is chosen at random from $\{0,1\}^n$ resp. y from $\{0,1\}^{\ell(n)}$. Pseudorandom generators exist if and only if one-way functions exist [HILL]. A function ensemble with key space $K = \{K_n\}_{n\in\mathbb{N}}$, input length $\text{in}(n)$ and output length $\text{out}(n)$ is a sequence $F = \{F^{(n)}\}_{n\in\mathbb{N}}$ of function families $F^{(n)} = \{f_k\}_{k\in K_n}$ such that for any $k \in K_n$ the function f_k maps bit strings of length $\text{in}(n)$ to bit strings of length $\text{out}(n)$. A function ensemble is *polynomial-time computable* if the length of the keys of $K = \{K_n\}$ and $\text{in}(n)$ are bounded by some polynomial in n and if there exists a polynomial-time algorithm Eval such that $\mathsf{Eval}(k, x) = f_k(x)$ for all n, $k \in K_n$ and $x \in \{0,1\}^{\text{in}(n)}$. In the sequel we denote by $\mathcal{R} = \{\mathcal{R}^{(n)}\}_{n\in\mathbb{N}}$ the function ensemble that contains all functions $g : \{0,1\}^{\text{in}(n)} \to \{0,1\}^{\text{out}(n)}$; here $\text{in}(n)$ and $\text{out}(n)$ and therefore the key space of $\mathcal{R}^{(n)}$ will be understood from the context. A polynomial-time computable function ensemble F (with key space K and input/output length $\text{in}(n)$ and $\text{out}(n)$) is a *pseudorandom function ensemble* (PRF ensemble) if for any probabilistic polynomial-time algorithm D, called the distinguisher, the advantage $\left|\text{Prob}\left[D^f(1^n) = 1\right] - \text{Prob}\left[D^g(1^n) = 1\right]\right|$ is negligible, where f is chosen at random from $F^{(n)}$ (by selecting a random key from K_n) and g is a random function of $\mathcal{R}^{(n)}$ (where each function in $\mathcal{R}^{(n)}$ has input/output length $\text{in}(n)$ and $\text{out}(n)$). A PRF ensemble F with key space K and input/output length $\text{in}(n) = \text{out}(n)$ is called a *pseudorandom permutation ensemble* (PRP ensemble) if f_k is a permutation for any key $k \in K_n$ and the advantage $\left|\text{Prob}\left[D^f(1^n) = 1\right] - \text{Prob}\left[D^g(1^n) = 1\right]\right|$ is negligible for any probabilistic polynomial-time algorithm D, where f is a random function of $F^{(n)}$ and g is a random permutation with input/output length $\text{in}(n) = \text{out}(n)$. A PRP ensemble F is said to be a *strong* PRP ensemble if it even holds that $|\text{Prob}[D^{f,f^{-1}}(1^n) = 1] - \text{Prob}[D^{g,g^{-1}}(1^n) = 1]|$ is negligible for any probabilistic polynomial-time algorithm D.

Pseudorandom function ensembles can be constructed from any pseudorandom generator via the GGM-tree design [GGM86]. Let G denote a length-doubling pseudorandom generator, i.e., with output length $\ell(n) = 2n$; such generators can be constructed from any pseudorandom generators by modifying the output length. Let $G^0(x)$ resp. $G^1(x)$ denote the left and right half of $G(x)$ and define the function ensemble F with key space $K_n = \{0,1\}^n$ and input/output length $\text{in}(n) = \text{out}(n) = n$ by $f_k(x) = G^{x_n}(\cdots G^{x_2}(G^{x_1}(k))\cdots)$. Here, $x_1, \ldots, x_n \in \{0,1\}$ and $x = x_1; \cdots; x_n$ is the concatenation of $x_1, \ldots, x_n$. The function f_k can be described by a binary tree of depth n where the root is labeled with k and each left (right) child of a node v is labeled with $G^0(\text{label}(v))$

resp. $G^1(\text{label}(v))$. A value $x \in \{0,1\}^n$ then determines a path from the root to some leaf and the function value $f_k(x)$ equals the label of this leaf. Goldreich et al. [GGM86] prove that the derived ensemble F is pseudorandom.

A PRF tribe function ensemble with key space $K = \{K_n\}_{n\in\mathbb{N}}$ and tribe key space $T = \{T_n\}_{n\in\mathbb{N}}$ is a function ensemble $F = \{\{F_t^{(n)}\}_{t\in T_n}\}_{n\in\mathbb{N}}$ of function families $F_t^{(n)} = \{f_k^t\}_{k\in K_n}$ such that $\{F_{t_n}^{(n)}\}_{n\in\mathbb{N}}$ is a PRF ensemble for any sequence $\{t_n\}_{n\in\mathbb{N}}$, $t_n \in T_n$ of tribe keys, and such that for a randomly chosen tribe key $t \in T_n$ the probability that there exist $x \in \{0,1\}^{\text{in}(n)}$, $k, k' \in K_n$ with $k \neq k'$ and $f_k^t(x) = f_{k'}^t(x)$ is at most 2^{-n}. The latter property is called (statistical) collision resistance.

3 Constructing PRF Tribe Ensembles

We first show how to construct an PRF ensemble F^{bind} such that $f_k^{\mathsf{bind}}(1^n) \neq f_{k'}^{\mathsf{bind}}(1^n)$ for keys $k \neq k'$. Put differently, the function value at 1^n commits to the key. We therefore say that this ensemble *binds the key* (for a fixed value) because once we have seen the value at 1^n one cannot later pretend to have used another key. Obviously, we can also take any other fixed value x_0 instead of 1^n by setting $f_k^*(x) = f_k^{\mathsf{bind}}(x \oplus x_0 \oplus 1^n)$. We then use such a fixed-value-key-binding PRF ensemble to derive a pseudorandom function (with tribe key t) where $f_k^t(x) \neq f_{k'}^t(x)$ for any $x, k \neq k'$ with probability $1 - 2^{-n}$ over the choice of t. This is achieved by using Naor's idea as explained in the introduction. We can even modify the construction to obtain a *key-binding-and-invertible* pseudorandom function that binds the key and can be efficiently inverted given the secret key. Particularly, this implies that $f_k^t(x) \neq f_{k'}^t(x')$ for $(k,x) \neq (k',x')$ with probability $1 - 2^{-n}$, i.e., the function binds the key *and* the preimage with high probability. This somewhat weaker property can also be derived extending the universal hash function $\text{XOR}(t, I_k)$ to take arguments x and I_k instead of I_k. We discuss this construction at the end of the section. However, it is not clear that this solution is efficiently invertible using the key, a requirement that we need in Section 4 applying our construction to private-key encryption.

3.1 A Fixed-Value-Key-Binding PRF Ensemble

Clearly, a pseudorandom function ensemble with $f_k(1^n) \neq f_{k'}(1^n)$ for $k \neq k'$ can be derived via the GGM-construction using a length-doubling pseudorandom generator G which is one-to-one on the right half. In this case, the function value at 1^n is $G^1(\cdots G^1(k))$ and since G^1 is one-to-one this yields different values for different keys. According to a result by Yao [Y82] such a pseudorandom generator G where G^1 is one-to-one can be constructed from any one-way permutation g by setting

$$G(x) = B(x); B(g(x)); \cdots; B(g^{|x|-1}(x)); g^{|x|}(x)$$

Here, $g^i(x) = g(g^{i-1}(x))$ and $g^1(x) = g(x)$ and B denotes some hardcore predicate of g. Obviously, $G^1(x) = g^{|x|}(x)$ is one-to-one (in fact, it is a permutation).

Another construction of fixed-value-key-binding ensembles was proposed by the authors of [CMR98] after presenting the GGM-based approach to them. The advantage is that we use the underlying pseudorandom function as a black box and merely add the length-doubling generator G (with G^1 being one-to-one) on. Particularly, instead of using the GGM-construction one can start with any PRF ensemble. For instance, more efficient constructions of PRF ensembles based on synthesizers [NR95] resp. on the Decisional-Diffie-Hellman assumption [NR97] suffice. In practice, one can also use appropriate candidates like the forthcoming AES.

So let F^{start} be an arbitrary PRF ensemble (the starting point). For simplicity, we suppose that each function f_k^{start} of $F^{\mathsf{start},(n)}$ maps n bits to n bits and that the key length equals n, too. We discuss below how to patch other cases. Set $k^b = G^b(k)$ for $b \in \{0,1\}$ and define the functions of $F^{\mathsf{bind},(n)}$ by

$$f_k^{\mathsf{bind}}(x) = \begin{cases} k^1 & \text{if } x = 1^n \\ f_{k^0}^{\mathsf{start}}(x) & \text{else} \end{cases}$$

Proposition 1. *F^{bind} is a fixed-value-key-binding PRF ensemble.*

Proof. (Sketch) The proof follows by standard hybrid techniques. Given a distinguisher D^{bind} that distinguishes F^{bind} and $\mathcal{R}$ with advantage $\delta(n)$ for infinitely many n, we either obtain an algorithm that distinguishes the output of G from random bits with advantage $\delta(n)/2$ infinitely often or we derive an algorithm that distinguishes F^{start} and $\mathcal{R}$ with advantage $\delta(n)/2$ for infinitely many n. More precisely, let the advantage of D^{bind} be at least $\delta(n) \geq 1/\operatorname{poly}(n)$ for infinitely many n and fix such an $n \in \mathbb{N}$. We start by defining a modified ensemble F^{mod} of F^{start} and show that any distinguisher D^{mod} that distinguishes F^{mod} and $\mathcal{R}$ with advantage $\epsilon(n)$ yields a distinguisher D^{start} that distinguishes F^{start} and $\mathcal{R}$ with advantage $\epsilon(n)$. Define the functions $f_{k;k^*}^{\mathsf{mod}}$ of F^{mod} by choosing (in addition to k) a random key $k^* \in \{0,1\}^n$ and replacing the output of f_k^{start} at 1^n by the random value k^*, that is, $f_{k;k^*}^{\mathsf{mod}}(x) = f_k^{\mathsf{start}}(x)$ if $x \neq 1^n$ and $f_{k;k^*}^{\mathsf{mod}}(x) = k^*$ if $x = 1^n$. If there is a distinguisher D^{mod} with advantage $\epsilon(n)$ then we construct D^{start} as follows. D^{start} is given an oracle f either chosen at random from F^{start} or from $\mathcal{R}$. D^{start} simulates D^{mod} by choosing a random $k^* \in \{0,1\}^n$ and answering all oracle queries of D^{mod} about x with $f^{\mathsf{sim}}(x)$, where $f^{\mathsf{sim}(x)} = f(x)$ for $x \neq 1^n$ and $f^{\mathsf{sim}}(x) = k^*$ for $x = 1^n$. Finally, D^{start} outputs whatever D^{mod} outputs. If f is chosen from F^{start} then f^{sim} is a random function of F^{mod}. Conversely, if f is a truly random function then so is f^{sim} because k^* is chosen at random. Hence, the advantage of D^{start} equals the advantage of D^{mod} and F^{mod} constitutes a pseudorandom function ensemble if F^{start} does.

Next we show that the distinguisher D^{bind} contradicts the pseudorandomness of G or F^{mod}. Consider the following experiments:

- given an input $y \in \{0,1\}^{2n}$ where either $y = G(k)$ for random $k \in \{0,1\}^n$ or y is uniformly chosen from $\{0,1\}^{2n}$, we simulate D^{bind} answering all oracle queries x of D^{bind} by computing $f_y^{\mathsf{mod}}(x)$. Finally, output whatever D^{bind}

outputs. If $y = G(k)$ then $f_y^{\mathsf{mod}}(x) = f_k^{\mathsf{bind}}(x)$; otherwise the simulated oracle corresponds to a random function f_{k,k^*}^{mod} of F^{mod}. Thus, the advantage in distinguishing a random value from the pseudorandom generator's output equals the advantage of D^{bind} distinguishing $F^{\mathsf{bind},(n)}$ and $F^{\mathsf{mod},(n)}$.

- given an oracle f from $F^{\mathsf{mod},(n)}$ or $\mathcal{R}^{(n)}$, simulate D^{bind} by answering each query about x with $f(x)$. Give the same output as D^{bind} does.

By assumption, for at least one of the experiments the advantage is at least $\delta(n)/2$. But this contradicts the pseudorandomness of G or F^{mod} (and therefore F^{start}). Obviously, F^{bind} binds the key for the fixed value 1^n because G^1 is one-to-one. $\square$

Though F^{start} might be a pseudorandom permutation ensemble, F^{bind} does not inherit this property in general. However, a slight modification works: swapping the values that map to k^1 and $f_{k^0}^{\mathsf{start}}(1^n)$ we let

$$f_k^{\mathsf{bind}}(x) = \begin{cases} k^1 & \text{if } x = 1^n \\ f_{k^0}^{\mathsf{start}}(1^n) & \text{if } f_{k^0}^{\mathsf{start}}(x) = k^1 \text{ and } x \neq 1^n \\ f_{k^0}^{\mathsf{start}}(x) & \text{else} \end{cases} \tag{1}$$

It is easy to see that f_k^{bind} is a permutation if $f_{k^0}^{\mathsf{start}}$ is. Moreover, the inverse of f_k^{bind} is efficiently computable (given the key k) if $f_{k^0}^{\mathsf{start}}$ has this property. We remark that every PRF ensemble can be turned into a PRP ensemble [LR88]; see [NR99] for recent results. Yet, using the Luby-Rackoff transformation, the key length of the derived permutation grows. This can be handled by stretching the output length of the generator G accordingly; it suffices that G is one-to-one on the bits that replace the output at 1^n. In particular, if the output length of $f_{k^0}^{\mathsf{start}}$ is smaller than right half of $G(k)$ then we can first stretch the output of $f_{k^0}^{\mathsf{start}}$ at the cost of decreasing the input length slightly. We will use this technique in the next section, too, so we omit further details here. The proof that the ensemble F^{bind} defined by equation (1) is pseudorandom is similar to the proof of Proposition 1. It is also easy to show that F^{bind} is a *strong* PRP ensemble if F^{start} is.

Proposition 2. *If F^{start} is a [strong] PRP ensemble then F^{bind} as defined in equation* (1) *is a fixed-value-key-binding [strong] PRP ensemble.*

We remark that once the key is generated (by evaluating the pseudorandom generator) computing $f_k^{\mathsf{bind}}(x)$ is as fast as computing $f_k^{\mathsf{start}}(x)$. Particularly, f_k^{start} may be any fast practical pseudorandom function candidate. In contrast, using the GGM-based approach we have to apply n times a pseudorandom generator which is one-to-one on the right half, e.g., based on a number-theoretic one-way permutation like RSA.

3.2 PRF Tribe Ensembles from Key-Binding PRF Ensembles

Let F^{bind} be a fixed-value-key-binding PRF ensemble (for the value 1^n). In another intermediary step we define a PRF ensemble F^{stretch} that has input length $n-3$, but stretches the output length to $5n$. Define the functions $f_k^{\mathsf{stretch}} : \{0,1\}^{n-3} \to \{0,1\}^{5n}$ by

$$f_k^{\mathsf{stretch}}(x) = f_k^{\mathsf{bind}}(x000); \cdots ; f_k^{\mathsf{bind}}(x011); f_k^{\mathsf{bind}}(x111)$$

Obviously, F^{stretch} is a PRF ensemble if F^{bind} is. Also note that computing f_k^{stretch} takes at most five evaluations of f_k^{bind}; but due to the common prefix one might not need to carry out all evaluations of f_k^{bind} from scratch and save time.

Now we are able to define our tribe ensemble F of functions $f_k^t : \{0,1\}^{n-3} \to \{0,1\}^{5n}$. The tribe key $t = (t_1, \ldots, t_n)$ consists of n uniformly and independently chosen values $t_i \in \{0,1\}^{4n} \times \{0^n\}$, i.e., t_i is a random $4n$-bit string filled up with 0-bits. Denote

$$I_k = f_k^{\mathsf{bind}}(1^n) = \text{rightmost } n \text{ bits of } f_k^{\mathsf{stretch}}(1^{n-3})$$

and let

$$\mathrm{XOR}(t, I_k) = \bigoplus_{i\text{-th bit}(I_k)=1} t_i$$

Then we set

$$f_k^t(x) = f_k^{\mathsf{stretch}}(x) \oplus \mathrm{XOR}(t, I_k) \tag{2}$$

Note that once k and t are chosen, $\mathrm{XOR}(t, I_k)$ is also fixed. Therefore, evaluating f_k^t at some point x is quasi as efficient as computing $f_k^{\mathsf{stretch}}(x)$. The proof that F is pseudorandom *for any sequence of tribe keys* is given below. We stress that the pseudorandomness of F does not depend on the random choice of the tribe key. See the discussion in [CMR98]. Also note that if f_k^{bind} is one-to-one (e.g., a permutation) then $f_k^{\mathsf{stretch}}(x) \neq f_k^{\mathsf{stretch}}(x')$ resp. $f_k^t(x) \neq f_k^t(x')$ for $x \neq x'$.

Proposition 3. *F is a PRF ensemble for any sequence of tribe keys.*

Proof. (Sketch) The proof follows by standard simulation arguments. Given an adversary D that distinguishes a random function of F and a randomly chosen function from the ensemble $\mathcal{R}$ we obtain a distinguisher D^{stretch} that distinguishes F^{stretch} and $\mathcal{R}$ with the same advantage. Note that both D and D^{stretch} are given an arbitrary tribe key t as input. For a function $f : \{0,1\}^{n-3} \to \{0,1\}^{5n}$ let $f^{\mathsf{sim}}(x) = f(x) \oplus \mathrm{XOR}(t, I)$, where I denotes the rightmost n bits of $f(1^{n-3})$. D^{stretch} simulates D by answering all oracle queries x with $f^{\mathsf{sim}}(x)$, where the underlying oracle f of D^{stretch} is chosen from $F^{\mathsf{stretch},(n)}$ or $\mathcal{R}^{(n)}$. If f is chosen at random from F^{stretch} then f^{sim} is a random function of F. Assume that f is a random function of $\mathcal{R}^{(n)}$. It is easy to see that in this case any value $f^{\mathsf{sim}}(x)$ is distributed independently of the other function values. Hence, it suffices to show that $\mathrm{Prob}_f\left[f^{\mathsf{sim}}(x) = y\right] = 2^{-5n}$ for any x, y. This is clear for $x \neq 1^{n-3}$. Consider the case $x = 1^{n-3}$. The rightmost n bits of $f(x)$ are random bits and the rightmost n bits of $\mathrm{XOR}(t, I)$ equal 0^n. Hence, with probability 2^{-n} we have equality on these bits. The leftmost $4n$ bits of $f(x)$ are random bits that are independent of the other n bits. Therefore, the probability that these bits equal the leftmost $4n$ bits of $y \oplus \mathrm{XOR}(t, I)$ is 2^{-4n} and both probabilities multiply due to the independence. $\square$

Recall that a PRF tribe ensemble is collision-resistant (in a statistical sense) if there do not exist x and k, k' such that $k \neq k'$ and $f_k^t(x) = f_{k'}^t(x)$ except with exponentially small probability (over the random choice of the tribe key). In our case, we have $I_k \neq I_{k'}$ for $k \neq k'$ and a collision

$$f_k^t(x) = f_k^{\mathsf{stretch}}(x) \oplus \mathrm{XOR}(t, I_k) = f_{k'}^{\mathsf{stretch}}(x) \oplus \mathrm{XOR}(t, I_{k'}) = f_{k'}^t(x)$$

implies

$$f_k^{\mathsf{stretch}}(x) \oplus f_{k'}^{\mathsf{stretch}}(x) = \mathrm{XOR}(t, I_k) \oplus \mathrm{XOR}(t, I_{k'}) = \mathrm{XOR}(t, I_k \oplus I_{k'})$$

Because $I_k \oplus I_{k'} \neq 0^n$, the value $\mathrm{XOR}(t, I_k \oplus I_{k'})$ is uniformly distributed in $\{0,1\}^{4n} \times \{0^n\}$ for fixed $x, k \neq k'$ and random t. By the union bound we conclude that

$$\mathrm{Prob}_t\left[\exists\, x, k \neq k' \ \text{ s.t. } \ f_k^t(x) = f_{k'}^t(x)\right] \leq 2^{3n-3} \cdot 2^{-4n} \leq 2^{-n}$$

Thus we obtain:

Theorem 1. *The ensemble F defined by equation* (2) *is a PRF tribe ensemble.*

Clearly, we can lower the error probability of the collision resistance. For example, to achieve an error of 2^{-4n} we extend f_k^{stretch} to $8n$ bits output and choose the t_i's at random from $\{0,1\}^{7n} \times \{0^n\}$. If, in addition to an extended output length of at least $6n$ bits, we use a pseudorandom permutation F^{start} then we derive a pseudorandom function tribe ensemble F such that $f_k^t(x) \neq f_{k'}^t(x')$ for $(k,x) \neq (k',x')$ with probability at least $1 - 2^{-n}$ (taken over the choice of the tribe key only) and which is efficiently invertible given the secret key (for all possible tribe keys); to invert a value $y = f_k^t(x)$ invert the rightmost n bits of y under the starting pseudorandom function to obtain $x111$ and therefore x (note that the rightmost n bits of $\mathrm{XOR}(t, I_k)$ equal 0^n). We call such an ensembles key-binding-and-invertible. Observe that the key-and-preimage-binding property alone can be achieved by taking output length $8n$ bits, choosing $2n-3$ strings t_i from $\{0,1\}^{7n} \times \{0^n\}$ and letting $\mathrm{XOR}(t, I_k, x)$ be the exclusive-or of the t_i's for which the i-th bit of $I_k; x$ equals 1.

4 Committing and Key-Hiding Private-Key Encryption

A well-known private-key encryption scheme based on PRF ensembles is given by $\mathsf{Enc}_k(m,r) = (r, f_k(r) \oplus m)$, where k is the secret key, m is the message and r is chosen at random. To decrypt a pair (r,c) compute $m = \mathsf{Dec}_k(r,c) = f_k(r) \oplus c$. This encryption scheme is not comitting in general, i.e., for an encryption (r,c) there might exist (k,m), (k',m') with $m \neq m'$ and $\mathsf{Enc}_k(m,r) = (r,c) = \mathsf{Enc}_{k'}(m',r)$. Conversely, we call a cryptosystem committing if for each ciphertext c there exists a unique message m such that c must have been derived by applying the encryption algorithm to m — this holds independently of the choice of the secret key and the coin tosses used during the encryption process.

Before presenting the formal definition of committing schemes we sketch the definition of a private-key cryptosystem. A private-key encryption scheme is a triple $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ of probabilistic polynomial-time algorithms such that

– KGen on input 1^n generates a random key k,
– Enc on input 1^n, key k, message m (of some appropriate length) and randomness r outputs a ciphertext $c = \mathsf{Enc}(1^n, k, m, r)$,
– $\mathsf{Dec}(1^n, k, \mathsf{Enc}(1^n, k, m, r)) = m$.

Wlog. we assume that 1^n is recoverable from k and therefore write $\mathsf{Enc}(k, m, r)$ or $\mathsf{Enc}_k(m, r)$ instead of $\mathsf{Enc}(1^n, k, m, r)$. Similarly for Dec.

Definition 1 (Committing Private-Key Encryption Scheme). *A private-key encryption scheme* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is called committing if for any key k, message m, randomness r and encryption $c = \mathsf{Enc}_k(m, r)$ there do not exist k', m', r' such that $m \neq m'$ and $\mathsf{Enc}_{k'}(m', r') = \mathsf{Enc}_k(m, r)$.*

Using a fixed-value-key-binding PRF ensemble the obvious solution $\mathsf{Enc}_k(m, r) = (f_k(1^n), r, f_k(r) \oplus m)$ works. This correpsonds to the case that one appends the same commitment of the key to each encryption. The drawback of this solution is that an eavesdropper knows whenever the parties change the secret key. In some settings hiding this fact might be crucial. For instance, if one party sends the new secret key by encrypting it with the current one, then breaking this encryption by an exhaustive search makes all the following messages visible to the adversary. We can overcome this disadvantage by computing a new commitment for each encryption, say, by applying the key-binding-and-invertible PRF tribe ensemble of Section 3. But before presenting our committing and key-hiding scheme we formalize the notion of a key-hiding scheme. Let $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a private-key encryption scheme and D be a probabilistic polynomial-time algorithm. We consider two experiments. In the first experiment, we independently execute $\mathsf{KGen}(1^n)$ twice to obtain two keys k, k'. D is given 1^n as input and is allowed to query the probabilistic oracles Enc_k and $\mathsf{Enc}_{k'}$ in the following way: In the first part, D is allowed to obtain encryptions of messages of its choice by querying the oracle Enc_k. Then it passes a message SWITCH to the oracle Enc_k. It continues to query for messages of its choice, but this time the answers are given by the second oracle $\mathsf{Enc}_{k'}$. Finally, D outputs a bit, denoted $D^{\mathsf{Enc}_k, \mathsf{Enc}_{k'}}(1^n)$, and stops. The second experiment differs only in the way the oracles are initialized. This time we let $k' = k$, i.e., we do not change the keys. Denote by $D^{\mathsf{Enc}_k, \mathsf{Enc}_k}(1^n)$ the output.

Definition 2 (Key-Hiding Private-Key Encryption Scheme). *A private-key encryption scheme* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be key-hiding if for any probabilistic polynomial-time algorithm D the value* $\left|\mathrm{Prob}\left[D^{\mathsf{Enc}_k, \mathsf{Enc}_{k'}}(1^n) = 1\right] - \mathrm{Prob}\left[D^{\mathsf{Enc}_k, \mathsf{Enc}_k}(1^n) = 1\right]\right|$ *is negligible in n.*

Actually, every secure[3] scheme should “hide” the key, i.e., it should not reveal the key. Otherwise it can be easily broken. However, Definition 2 demands even more. For instance, an encryption scheme where each encryption leaks the Hamming weight of the key with some probability that is not negligible does not hide the key as defined above. Yet, the scheme may be secure.

[3] Here, security does not refer to any formal definition. It is used in a rather liberal sense.

We remark that we do not grant D access to the decryption oracles Dec_k and $\mathsf{Dec}_{k'}$, respectively. Otherwise D could distinguish both cases easily: D encrypts some message m with the first oracle, sends SWITCH and tries to decrypt with the second decryption oracle; this only yields m again if the keys have not changed.

Next we define our committing and key-hiding private-key encryption scheme $(\mathsf{KGen}^{\mathsf{com}}, \mathsf{Enc}^{\mathsf{com}}, \mathsf{Dec}^{\mathsf{com}})$. Let F be a PRF tribe ensemble derived by the technique of Section 3.2 from a key-binding-and-invertible ensemble F^{bind}. We assume that some trusted party chooses a random tribe key t and publishes it or sends it to the participating parties, respectively. Hence, we do not achieve the committing property of Definition 1 perfectly, but only with exponentially small error probability. Abusing notations we will also call this derived scheme committing. Algorithm $\mathsf{KGen}^{\mathsf{com}}(1^n)$ selects a random $k \in K_n$. Let $\mathsf{Enc}_k^{\mathsf{com}}(m, r) = (f_k^t(r), r \oplus m)$ where $m, r \in \{0,1\}^{n-3}$. To decrypt a pair (y, c) compute r from the rightmost n bits of y by applying the inverse of f_k^{bind}. Finally, recover m by $m = r \oplus c$.

Proposition 4. *The encryption scheme* $(\mathsf{KGen}^{\mathsf{com}}, \mathsf{Enc}^{\mathsf{com}}, \mathsf{Dec}^{\mathsf{com}})$ *is a committing and key-hiding encryption scheme.*

Proof. (Sketch) It remains to show that the scheme is key-hiding. But this follows directly from the pseudorandomness of F^{bind}. □

It is quite easy to see that this scheme is polynomially secure as defined in [GM84]. We sketch this and other security notions in Appendix B. In fact, it is not hard to show either that it is even secure against lunchtime attacks [NY90] if we use a strong PRP:

Proposition 5. *If* F^{bind} *is a strong PRP ensemble, then the encryption scheme* $(\mathsf{KGen}^{\mathsf{com}}, \mathsf{Enc}^{\mathsf{com}}, \mathsf{Dec}^{\mathsf{com}})$ *is a committing and key-hiding private-key encryption which is secure against lunchtime attacks.*

Proof. (Idea) Otherwise we derive a contradiction to the pseudorandomness of F^{bind} (which also contradicts the pseudorandomness of the tribe ensemble F). A distinguisher D that is given oracle access to a function f and its inverse function f^{-1} can simulate the lunchtime attacker by choosing the r's at random and computing the encryption via the oracle f, and decrypt by using f^{-1}. If there exist a successful lunchtime attacker then we obtain an algorithm that distinguishes at least one hybrid pair of the following distributions successfully:

1. answering all encryption and decryption queries with f, f^{-1}, where f randomly chosen from F^{bind}, and encrypting the challenge message m_0
2. answering the queries with a truly random function pair f, f^{-1} from the set $\mathcal{R}$ of all permutations and encrypting m_0
3. answering all queries with a random function pair f, f^{-1} from $\mathcal{R}$ and encrypting m_1
4. answering all queries with a random function pair f, f^{-1} from F and encrypting m_1

If the first and second distribution or the third and fourth distribution can be distinguished significantly then we derive a contradiction to the pseudorandomness of F^{bind}. Assume that one can distinguish the second and third distribution successfully. Observe that the challenge (y, c) for m_0 resp. m_1 leaves two possibilities r_0, r_1 for r: either $r_0 = m_0 \oplus c$ or $r_1 = m_1 \oplus c$. If both values have not appeared among the previous queries (which happens with probability at least $1 - 2 \cdot \operatorname{poly}(n) \cdot 2^{-n+3}$) then y (basically) is a random value and reveals no information about r and therefore the encrypted message. Hence, there is no possibility to distinguish encryptions of m_0 and m_1 with probability at least $1/\operatorname{poly}(n)$ and this case cannot occur. □

The encryption scheme can be easily broken with a chosen ciphertext and plaintext attacks (see [RS91] or Appendix B) because given a ciphertext (y, c) the adversary can query the decryption oracle for $(y, c \oplus 1^{|c|})$ and easily recover m from the answer. Using an idea of Bellare and Rogaway [BR93] we can turn the scheme above into an encryption scheme $(\mathsf{KGen}^{\mathsf{ccp}}, \mathsf{Enc}^{\mathsf{ccp}}, \mathsf{Dec}^{\mathsf{ccp}})$ which is secure against chosen ciphertext and plaintext attacks. To this end, we let

$$\mathsf{Enc}_k^{\mathsf{ccp}}(r, m) = (f_k^t(r; m), r \oplus m)$$

for $m, r \in \{0,1\}^{n/2-1}$. Defining $\mathsf{Dec}^{\mathsf{ccp}}$ is straightforward. Loosely speaking, appending m to the argument r of the pseudorandom function serves as a proof that one knows the values r, m explicitely.

Proposition 6. *If F^{bind} is strong PRP, then the committing and key-hiding private-key encryption scheme $(\mathsf{KGen}^{\mathsf{ccp}}, \mathsf{Enc}^{\mathsf{ccp}}, \mathsf{Dec}^{\mathsf{ccp}})$ is secure against chosen ciphertext and plaintext attacks.*

Proof. (Idea) Letting the adversary encrypt messages of its choice after receiving the challenge does not endanger our scheme. It suffices to show that the decryption oracle does not help significantly. The simulator D of the ciphertext attacker returns Error whenever the attacker puts a decryption query (y, c) for some value y that has not been appeared previously as part of an encryption output, or if c does not equal $r \oplus m$ for the values r, m which have been used when computing y previously. Else it answers with the known value m. This simulation fails with negligible probability only, since otherwise we could construct an algorithm that successfully distinguishes the output of a pseudorandom function for a "new" input value and a truly random string. But this would contradict the pseudorandomness of the ensemble [GGM86]. Let us consider this in more detail. Assume that with probability $1/\operatorname{poly}(n)$ the attacker puts a query (y, c) such that y has not been output by the encryption oracle so far, and such that c is correct. We simply try to guess this query (among the at most polynomial many queries) and then pass y as a challenge and receive either $f^{-1}(y)$ for the oracle pair f, f^{-1} chosen at random from F^{bind}, or a random string z. We are supposed to tell both cases apart. By assumption, c equals the xor of the left and right half of $f^{-1}(y)$ with probability $1/\operatorname{poly}(n)$. But the probability that it equals the corresponding xor of a random string z is negligible. Therefore, we can distinguish both cases with probability $1/\operatorname{poly}(n)$, which contradicts the pseudorandomness of f^{-1}. Now, given that this simulation succeeds, we obtain the desired result by applying the same argument of the proof of Proposition 5.

□

Recently, Dolev et al. [DDN99] and Bellare et al. [BDPR98] showed that (semantic) security against chosen ciphertext and plaintext attacks implies non-malleability. Hence, our scheme is non-malleable as well. In fact, our construction gives an alternative to the non-malleable private-key scheme presented in [DDN99]. The construction there is similar, but needs two pseudorandom function evaluations with two independent secret keys. Nevertheless, the construction there does not require the pseudorandom functions to be strong pseudorandom permutations.

Acknowledgements

We are grateful to Ran Canetti, Daniele Micciancio and Omer Reingold for discussing our ideas and proposing the more practical construction of F^{bind} in Section 3.1. We also thank the anonymous referees of Eurocrypt'99 for their comments.

References

[BDJR97] M.Bellare, A.Desai, E.Jokipii, P.Rogaway: A Concrete Security Treatment of Symmetric Encryption, *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 394–403*, 1997.

[BDPR98] M. Bellare, A.Desai, D.Pointcheval, P.Rogaway: Relations Among Notions of Security for Public-Key Encryption Schemes, *Crypto '98, Lecture Notes in Computer Science, Vol. 1462, Springer-Verlag*, 1998.

[BR93] M.Bellare, P.Rogaway: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols, *First ACM Conference on Computer and Communications Security*, 1993.

[C97] R.Canetti: Towards Realizing Random Oracles: Hash Functions that Hide All Partial Information, *Crypto '97, Lecture Notes in Computer Science, Vol. 1294, Springer-Verlag, pp. 455–469*, 1997.

[CDNO97] R.Canetti, C.Dwork, M.Naor. R.Ostrovsky: Deniable Encryption, *Crypto '97, Lecture Notes in Computer Science, Vol. 1294, Springer-Verlag, pp. 90–104*, 1997.

[CFGN96] R.Canetti, U.Feige, O.Goldreich, M.Naor: Adaptively Secure Multi-Party Computation, *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC), pp. 639–648*, 1996.

[CG96] R.Canetti, R.Gennaro: Incoercible Multiparty Computation, *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 504–513*, 1996.

[CMR98] R.Canetti, D.Micciancio, O.Reingold: Perfectly One-Way Probabilistic Hash Functions, *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC)*, 1998.

[CW79] L.Carter, M.Wegman: Universal Classes of Hash Functions, *Journal of Computer and System Science, vol. 18, pp. 143–154*, 1979.

[DDN99] D.Dolev, C.Dwork, M.Naor: Non-Malleable Cryptography, *submitted journal version; a preliminary version appearded in Proceedings of the 23rd Annual ACM Symposium on the Theory of Computing (STOC) in 1991*, 1999.

[GGM86] S.Goldwasser, O.Goldreich, S.Micali: How to Construct Random Functions, *Journal of ACM, vol. 33, pp. 792–807*, 1986.

[G95] O.Goldreich: Foundations of Cryptography (Fragments of a Book), *Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel*, 1995.

[GL89] O.Goldreich, L.Levin: A Hardcore Predicate for All One-Way Functions, *Proceedings of the 21st Annual ACM Symposium on the Theory of Computing (STOC), pp. 25–32*, 1989.

[GM84] S.Goldwasser, S.Micali: Probabilistic Encryption, *Journal of Computer and System Science, Vol. 28, pp. 270–299*, 1984.

[HILL] J.Hastad, R.Impagliazzo, L.Levin, M.Luby: Construction of a Pseudorandom Generator from any One-Way Function, *to appear in SIAM Journal on Computing, preliminary versions in STOC'89 and STOC'90*, 1989/90.

[CISI99] Interministerial Committee on the Information Society (CISI): Build a Legislative Framework to Protect Exchanges and Privacy, *available at* `http://www.internet.gouv.fr/english/textesref/cisigb/fiche1gb.htm`, January 19, 1999.

[LR88] M.Luby, C.Rackoff: How to Construct Pseudorandom Permutations from Pseudorandom Functions, *SIAM Journal on Computing, Vol. 17, pp. 373–386*, 1988.

[N91] M.Naor: Bit Commitment Using Pseudo-Randomness, *Journal of Cryptology, vol. 4, pp. 151–158*, 1991.

[NR95] M.Naor, O.Reingold: Synthesizers and Their Application to the Parallel Construction of Pseudorandom Functions, *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 170–181*, 1995.

[NR97] M.Naor, O.Reingold: Number-Theoretic Constructions of Efficient Pseudorandom Functions, *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 458–467*, 1997.

[NR99] M.Naor, O.Reingold: On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited, *Journal of Cryptology, vol. 12, no. 1, pp. 29–66*, 1999.

[NY90] M.Naor, M.Yung: Public-Key Cryptosystems Provably Secure Against Chosen Ciphertext Attacks, *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing (STOC), pp. 427–437*, 1990.

[RS91] C.Rackoff, D.Simon: Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attacks, *Crypto '91, Lecture Notes in Computer Science, Vol. 576, Springer-Verlag, pp. 433–444*, 1991.

[Y82] A.C.Yao: Theory and Application of Trapdoor Functions, *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 80–91*, 1982.

A The [CMR98] PRF Tribe Ensemble — in a Nutshell

We sketch the construction of PRF tribe ensembles from one-way permutations given in [CMR98]. See their paper for discussions and proofs. Let g' be a one-way permutation over $\{0,1\}^{6n}$ and assume that $g'(x;r) = g(x);r$ for $x, r \in \{0,1\}^{3n}$. Furthermore, we can assume wlog. that g has no cycles of length less than $12n$. Let p be a non-constant polynomial over $\mathrm{GF}[2^{6n}]$ and define a hardcore predicate $B_p : \{0,1\}^{6n} \to \{0,1\}$ of g' by the inner product $B_p(x;r) = p(x) \cdot r$ of $p(x), r \in \{0,1\}^{3n}$. Then, for any polynomial p, we construct a length-doubling pseudorandom generator by

$$G_p(x;r) = B_p(x;r); B_p(g(x);r); \cdots ; B_p(g^{6n-1}(x);r); g^{6n}(x); r$$

Denote by $G_p^0(x;r)$ and $G_p^1(x;r)$ the left and right half of $G_p(x;r)$. Additionally, we let $G : \{0,1\}^n \to \{0,1\}^{6n}$ denote an arbitrary pseudorandom generator which is one-to-one on the right half.

The tribe key t consists of n random, non-constant polynomials $p_1, \ldots, p_n$ of degree less than $6n$. Then let

$$f_k^t(x) = G_{p_n}^{x_n}(\cdots G_{p_1}^{x_1}(G(k)))$$

That is, f_k^t is a GGM-tree using pseudorandom generators based on the modified Goldreich-Levin hardcore predicate.

B Security Notions of Private-Key Encryption Schemes

In this section we recall the notions of polynomial security [GM84], security against lunchtimes attacks [NY90] resp. against chosen ciphertext and plaintext attacks [RS91]. See [BDJR97] for further security definitions for symmetric schemes. We refer the reader to [DDN99] for a definition of non-malleable schemes, a notion that turned out to be equivalent to security against chosen ciphertext and plaintext attacks.

Consider the following attack on a private-key cryptosystem. Let (F, D) be a pair of probabilistic polynomial-time algorithms. First, a secret key k is chosen according to $\mathsf{KGen}(1^n)$ and kept secret from F and D. Then the message finder F gets the input 1^n and outputs two messages m_0, m_1. Let $b \in \{0,1\}$. A ciphertext $c = \mathsf{Enc}_k(m_b, r)$ for randomness r is generated. Now D is given input 1^n, m_0, m_1 and c and is supposed to predict b, i.e., to distinguish encryptions of m_0 and m_1. Let $\delta_{F,D}^b(n)$ denote the probability that D outputs 1 if m_b is encrypted. The probability is taken over all random choices, including the internal coin tosses of F and D.

An encryption scheme is *polynomially secure* if D cannot distinguish an encryption of m_0 from an encryption of m_1 significantly. More formally, it is polynomially secure if for all (probabilistic polynomial-time) adversary pairs (F, D) the value $|\delta_{F,D}^0(n) - \delta_{F,D}^1(n)|$ is negligible in n.

A lunchtime attack is similar to the aforementioned attack, but F is also allowed to adaptively query the encryption/decryption oracle for plaintexts and ciphertexts of its choice before outputting m_0, m_1, and D is given the history of this query/answer sequence as additional input. An encryption scheme is *secure against lunchtime attacks* if it still holds that $|\delta^0_{F,D}(n) - \delta^1_{F,D}(n)|$ is negligible in n for all (probabilistic polynomial-time) adversary pairs (F, D).

A chosen ciphertext and plaintext attack is a lunchtime attack where D is also allowed to adaptively query the encryption/decryption oracle — though D is of course not allowed to decipher the challenge c. Again, an encryption scheme is *secure against chosen ciphertext and plaintext attacks* if $|\delta^0_{F,D}(n) - \delta^1_{F,D}(n)|$ is negligible in n for all (probabilistic polynomial-time) adversary pairs (F, D).