Equivalence of Uniform Key Agreement and Composition Insecurity[⋆]

Chongwon Cho[1] [⋆⋆], Chen-Kuei Lee[1], and Rafail Ostrovsky[2] [⋆⋆⋆]

[1] Department of Computer Science, UCLA
[2] Department of Computer Science and Mathematics, UCLA
{ccho,jcklee, rafail}@cs.ucla.edu

Abstract. We prove that achieving adaptive security from composing two general non-adaptively secure pseudo-random functions is impossible if and only if a uniform-transcript key agreement protocol exists.
It is well known that proving the security of a key agreement protocol (even in a special case where the protocol transcript looks random to an outside observer) is at least as difficult as proving $P \neq NP$. Another (seemingly unrelated) statement in cryptography is the existence of two or more non-adaptively secure pseudo-random functions that do not become adaptively secure under sequential or parallel composition. In 2006, Pietrzak showed that *at least one* of these two seemingly unrelated statements is true. Pietrzak's result was significant since it showed a surprising connection between the worlds of public-key (i.e., "cryptomania") and private-key cryptography (i.e., "minicrypt"). In this paper we show that this duality is far stronger: we show that *at least one* of these two statements must also be false. In other words, we show their *equivalence*.
More specifically, Pietrzak's paper shows that if sequential composition of two non-adaptively secure pseudo-random functions is not adaptively secure, then there exists a key agreement protocol. However, Pietrzak's construction implies a slightly stronger fact: If sequential composition does not imply adaptive security (in the above sense), then a *uniform-transcript* key agreement protocol exists, where by uniform-transcript we mean a key agreement protocol where the transcript of the protocol execution is indistinguishable from uniform to eavesdroppers. In this paper, we complete the picture, and show the reverse direction as well as a strong equivalence between these two notions. More specifically, as our main result, we show that if there exists *any* uniform-transcript key agreement protocol, then composition does not imply adaptive security. Our result holds for both parallel and sequential composition. Our implication holds based on virtually all known key agreement protocols, and can also be based on general complexity assumptions of the existence of dense trapdoor permutations.

[⋆] Full version appeared on ECCC (Report No.: TR09-108, 31st October 2009).
[⋆⋆] Supported in part by grants 0716835, 0716389, 0830803, 0916574.
[⋆⋆⋆] Supported in part by IBM Faculty Award, Xerox Innovation Group Award, the Okawa Foundation Research Award, Intel, Teradata, Lockheed-Martin, NSF grants 0716835, 0716389, 0830803, 0916574, BSF grant, and U.C. MICRO grant.

1 Introduction

One of the central questions in cryptography is the question of *composition*, which very broadly is the study of various ways to compose several basic primitives in a way that amplifies the hardness of the composed object. Naturally, this central question has received a lot of attention in various settings and we continue the study of this question here. More specifically, we investigate a question of whether a composition of pseudo-random functions, to be defined shortly, constitutes stronger security by utilizing the security of the component functions. We consider two very natural types of conventional compositions: a parallel composition with respect to Exclusive-Or (XOR) operation denoted by $\oplus$ and a sequential composition. Briefly, on input x in the domain of F and G, the parallel XOR-composition of two functions F and G is defined as $\mathsf{F}(x) \oplus \mathsf{G}(x)$. The sequential composition of F and G is defined as $\mathsf{G}(\mathsf{F}(x))$ (or $\mathsf{F}(\mathsf{G}(x))$).

Seemingly unrelated to the notion of security amplification via composition, there is the question of designing Key Agreement protocol. Recall that Key Agreement (KA) is a protocol that enables two parties to generate a secret string (also called key) by communicating with each other over an insecure channel in the presence of a eavesdropping adversary. Uniform-transcript key agreement (UTKA) is a strengthened version of key agreement in which messages between two parties are indistinguishable from uniform distribution by all probabilistic polynomial-time (PPT) adversaries. The reason why key agreement seems unrelated to the security of composition is that key agreement belongs to the world of public-key cryptography (also known as "cryptomania") whereas the security of composed pseudo-random functions rather belongs to the world of private-key cryptography (also known as "minicrypt"). For further discussion on cryptomania and minicrypt, see [4].

Now, let us recall briefly recall the definition of Pseudo-Random Functions (PRF) [2]. There are two notions of security of PRF: adaptive security and non-adaptive security. Intuitively, a (pseudo-random) function is said to be non-adaptively secure if the function is indistinguishable from a random function against all PPT adversaries that evaluate the function on inputs chosen independently of the function outputs, that is, chosen prior to PPT adversary learning any of the outputs. Adaptive security is a far stronger notion of security than non-adaptive security: a PRF is said to be adaptively secure if the function remains indistinguishable from random function against all PPT adversaries preparing the current query based on the outputs of the function on all previous queries. Clearly, adaptive security implies non-adaptive security.

We show that the equivalence between the impossibility of achieving adaptive security by composing general non-adaptively secure pseudo-random functions and the existence of uniform transcript key-agreement protocol. We note that our impossibility result holds not only for the case in which the non-adaptively-secure component functions are drawn from the different function families (also known as the general composition) but also for the case where the component functions are drawn from the same function family (also known as self-composition).

1.1 Related Work

There has been extensive research on relationship between the security of component functions and the security of their parallel or sequential composition. In the information theoretic context, Vaudenay [11] proved that if F is a pseudo-random permutation with security ϵ against any distinguisher making q (non-)adaptive queries, then the sequential composition of k F's has improved security $2^{k-1}\epsilon^k$ against a (non-)adaptive distinguisher. F only needs to be a function instead of a permutation for the same security in parallel composition. Luby and Rackoff [5] show the similar security amplification result in the computational context.

In the information theoretic setting, Maurer and Pietrzak [6] proved that composition of non-adaptive secure functions amplifies its security ϵ to security $2\epsilon(1+\ln(\epsilon^{-1}))$ against an adaptive distinguisher. In 2007, Maurer et al. improved this bound to 2ϵ [7].

Myers [8] showed that the existence of oracles relative to which there are non-adaptively secure permutations, but where the composition of such permutations fails to achieve adaptive security. Recently, Pietrzak [9] showed that the composition of non-adaptively secure functions does not imply adaptive security under the Decisional Diffie-Hellman (DDH) assumption. Pietrzak's more recent work [Pie06] showed that if sequential composition does not imply adaptive security, then there exists a key agreement protocol. Moreover, it turns out that Pietrzak's construction in [10] implies a slightly stronger result: that his key agreement protocol satisfies the property of uniform-transcript. Thus, we can restate the Pietrazak's result as follows:

Theorem 1. [10] *If sequential composition of pseudo-random functions is not adaptively secure, then there exists a UTKA.*

1.2 Our Results

Pietrzak's work left open the question of establishing the precise connection between the impossibility of adaptively secure composition and key agreement. Our main contribution is to establish sufficient and necessary conditions. In particular, we prove that the existence of UTKA implies the impossibility of obtaining an adaptively secure function from composing general non-adaptively secure functions. The main technique is the fully black-box construction of counter-example functions from UTKA. Therefore, our result holds with respect to any UTKA without relying on the actual code of the UTKA. We prove our result in both parallel and sequential compositions.

Theorem 2. *If there exists a UTKA, then parallel composition of non-adaptively secure pseudo-random functions does not imply a pseudo-random function with adaptive security.*

Theorem 3. *If there exists a UTKA, then sequential composition of non-adaptively secure pseudo-random functions does not imply a pseudo-random function with adaptive security.*

We also prove the analog of Pietrzak's Theorem 1 for parallel composition:

Theorem 4. *If a parallel composition of speudo-random functions is not adaptively secure, then there exists a UTKA.*

Putting all our results together with Theorem 1, we conclude the equivalence between the impossibility of adaptively secure composition and the existence of a uniform transcript key-agreement (both for parallel and sequential compositions). This is informally stated as follows.

Theorem 5. (MAIN) *The composition of two non-adaptively secure pseudo-random functions does not imply an adaptively secure pseudo-random function if and only if a UTKA exists.*

We emphasize that our main theorem holds regardless of whether PRFs being composed are taken from a single function family (called self-composition) or from two distinct function families (called general-composition). In particular, we show that the impossibility of secure general-compositions further implies the impossibility of secure self-compositions. The precise connection between the impossibility of adaptively secure composition and a UTKA protocol were not known prior to our work. We summarize these previously known results and our contributions in Fig. 1.

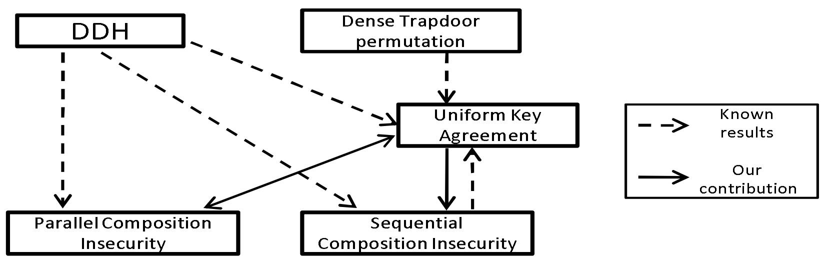
DDH
Dense Trapdoor permutation
Uniform Key Agreement
Known results
Our contribution
Parallel Composition Insecurity
Sequential Composition Insecurity

Fig. 1. Relationship between composition insecurity and other assumptions

Organization of the rest of the paper

In Section 2, we review all basic cryptographic notions and definitions. To build the intuition of our main construction, we first show in section 3 a high level outline of somewhat weaker result. In particular, we outline the analogue of Theorem 2 and Theorem 3 not assuming UTKA, but rather assuming the existence of a family of enhanced trapdoor permutations. We note that even this weaker variant of our main result is a generalization from the result by [9], which relies on a specific assumption (i.e., DDH assumption). In section 4 we proceed to give the intuition of our main result assuming UTKA. In section 5, we extend our main results to the one in the context of self-composition. We provide the complete constructions of our functions and full proofs of all theorems in [1].

2 Preliminaries

We let $n \in \mathbb{N}$ be a security parameter. An algorithm is considered efficient if its computation can be carried out by a PPT machine whose running time is expected polynomial in the input length. We use the notation $x \quad _{\$} \{0,1\}^n$ when string x is uniformly drawn from $\{0,1\}^n$. We omitted the rest of standard notations and (well-known) formal definitions. For those definitions, we refer the readers to [1].

3 Building intuition: Composition Insecurity vs. Dense Trapdoor Permutation

For gentle introduction to our main result, we first present a special case of our main result as an example – The existence of dense trapdoor permutation (DTP) implies the impossibility of achieving the adaptive security by composing (in a black-box way) non-adaptively secure pseudo-random functions. The main idea behind showing this, is that a family of DTPs is well-known to provide a 2-pass (uniform-transcript) key agreement.

3.1 Parallel Composition Insecurity from Dense Trapdoor Permutation

We construct two counter-example pseudo-random functions F and G which are secure against any PPT adversary non-adaptively. Then, we prove that their parallel composition is not secure against a particular sequence of four adaptive queries.

Intuitions of Parallel Composition of F and G We provide the high-level overview and intuition of our construction of pseudo-random functions F and G based on DTP, and show how to break the adaptive security of their parallel composition. The main technique of our constructions of counter-example functions is to design the functions to detect the adaptive query throughout the input and output behavior. In particular, F and G emulate a 2-pass key agreement protocol via adaptive inputs and outputs. Once F and G internally obtain a shared key, they generate outputs which hide a special relation with respect to the shared key. As we input these specially generated outputs to the parallel composition again, F and G retrieve the previously shared key and verify the special relation with respect to the shared key. Hence, function F and G are convinced that the queries must be indeed adaptively generated, and reveal their private keys through their outputs, which break their security.

Our counter-example functions F and G are both defined over $(\{0,1\}^n)^{2n+3}$. F and G hide the secret keys k_{F} and k_{G} respectively. P denotes an adaptively secure pseudo-random permutation. Let $(\mathsf{Gen}(\cdot), f, f^{-1})$ be a family of DTPs. r_{ij} and s_{ij} denote the ith pseudo-random string generated by F and G using their secret keys on jth input respectively. In addition, $\mathsf{Enc}_k(x)$ is defined to be a pseudo-random private-key encryption of x with respect to key k. Hence, we have $x = \mathsf{Dec}_k(\mathsf{Enc}_k(x))$.

We first define F and G on the first *fixed* adaptive query $Q_1 = (0^n, 0^n, \cdots, 0^n)$:

- F generates $2n+3$ pseudo-random strings $r^*, r_{21}, r_{31}, \cdots, r_{(2n+3)1}$ computed by $\mathsf{P}_{k_{\mathsf{F}}}(Q_1)$.
- G on input Q_1 uses its secret key to first compute sufficiently long pseudo-random string which is then used to compute DTP pair $(k,\ t_k)$: a pair of a DTP key k and its private trapdoor t_k by $\mathsf{Gen}(1^n)$ of DTP. G generates $2n+2$ pseudo-random strings $s_{21}, s_{31}, \cdots, s_{(2n+3)1}$ by $\mathsf{P}_{k_{\mathsf{G}}}(Q_1)$, then it outputs $(k, s_{21}, \cdots, s_{(2n+3)1})$.

We describe the outputs of F and G, and their parallel composition outputs below:

$$Q_1 \rightarrow \begin{bmatrix} \mathsf{F} \rightarrow (r^*, r_{21}, \cdots, r_{(2n+3)1}) \\ \mathsf{G} \rightarrow (k, s_{21}, \cdots, s_{(2n+3)1}) \end{bmatrix} \rightarrow (r^* \oplus k, r_{21} \oplus s_{21}, \cdots, r_{(2n+3)1} \oplus s_{(2n+3)1})$$

The second adaptive query is of the form $Q_2 = (u, 0^n, 0^n, \cdots, 0^n)$ where $u = r^* \oplus k$. We define F and G on Q_2 as follows.

- F first simulates the first-round of computation (by internally executing $\mathsf{P}_{k_{\mathsf{F}}}$ on the fixed query Q_1) to obtain r^*, then computes $u \oplus r^*$ which is equal to k; Now, F computes $2n+3$ pseudo-random strings $x_1, x_2, \cdots, x_n$ and $r_{(n+1)2}, r_{(n+2)2}, \cdots, r_{(2n+3)2}$ by $\mathsf{P}_{k_{\mathsf{F}}}(Q_2)$. F computes y_i by $f_k(x_i)$ for $1 \leq i \leq n$, then outputs $(y_1, \cdots, y_n, r_{(n+1)2}, \cdots, r_{(2n+3)2})$.
- G generates fresh pseudo-random strings $(s_{12}, s_{22}, \cdots, s_{(2n+3)2})$ computed by $\mathsf{P}_{k_{\mathsf{G}}}(Q_2)$.

We describe what both F and G output individually and the output of their parallel composition:

$$Q_2 \rightarrow \begin{bmatrix} \mathsf{F} \rightarrow (y_1, \cdots, y_n, r_{(n+1)2}, \cdots, r_{(2n+3)2}) \\ \mathsf{G} \rightarrow (s_{12}, \cdots, s_{n2}, s_{(n+1)2} \cdots, s_{(2n+3)2}) \end{bmatrix}$$

$$\rightarrow (y_1 \oplus s_{12}, \cdots, y_n \oplus s_{n2}, r_{(n+1)2} \oplus s_{(n+1)2}, \cdots, r_{(2n+3)2} \oplus s_{(2n+3)2})$$

We define the third adaptive query Q_3 to consist of the selected coordinates in the previous outputs such that $Q_3 = (y_1 \oplus s_{12},\ \cdots,\ y_n \oplus s_{n2}, r_{(n+1)2} \oplus s_{(n+1)2}, \cdots, r_{(2n)2} \oplus s_{(2n)2}, k \oplus r^*, 0^n, 0^n)$. On Q_3, we defined F and G as follows.

- F regenerates all the pseudo-random strings in the second round, $x_1, \cdots, x_n$, $r_{(n+1)2}, \cdots, r_{(2n+3)2}$ by $\mathsf{P}_{k_{\mathsf{F}}}(Q_2)$. Notice that Q_2 is $(k \oplus r^*, 0^n, \cdots, 0^n)$ where

F can obtain $k \oplus r^*$ from Q_3. F can compute $b_i =< x_i, r_{(n+i)2} >$ for all $1 \leq i \leq n$ and retrieve a shared key *sk* by letting $sk = b_1 b_2 \cdots b_n$. Now, F generates pseudo-random strings $r_{13}, r_{23}, \cdots, r_{(2n+3)3}$ by $\mathsf{P}_{k_\mathsf{F}}(Q_3)$ and encrypts r_{13} with the shared key as $\mathsf{Enc}_{sk}(r_{13})$. Finally, F outputs $(\mathsf{Enc}_{sk}(r_{13}), r_{13}, r_{23}, \cdots, r_{(2n+2)3})$.

– G regenerates $s_{12}, s_{22}, \cdots, s_{(2n)2}$ by $\mathsf{P}_{k_\mathsf{G}}(Q_2)$. G can obtain $y_1, \cdots, y_n, r_{(n+1)2}, \cdots, r_{(2n)2}$ as it cancels $s_{12}, s_{22}, \cdots, s_{(2n)2}$ out of the first $2n$ coordinates in Q_3. By using the inverse permutation $f_{t_k}^{-1}$ with respect to the trapdoor t_k, G can obtain x_i by computing $f_{t_k}^{-1}(y_i)$ for all i. Hence, G can compute $b_i =< x_i, r_i >$ for all i and retrieve the shared key *sk* by letting $sk = b_1 b_2 \cdots b_n$. Then, G generates pseudo-random strings $s_{13}, s_{23}, \cdots, s_{(2n+3)3}$ by $\mathsf{P}_{k_\mathsf{G}}(Q_3)$ and creates an encryption $\mathsf{Enc}_{sk}(s_{13})$. Finally, G outputs $(\mathsf{Enc}_{sk}(s_{13}), s_{13}, s_{23}, \cdots, s_{(2n+2)3})$.

Below we depict the individual outputs of F and G and the output of their parallel composition:

$$Q_3 \rightarrow \begin{bmatrix} \mathsf{F} \rightarrow (\mathsf{Enc}_{sk}(r_{13}), r_{13}, r_{23}, \cdots, r_{(2n+2)3}) \\ \mathsf{G} \rightarrow (\mathsf{Enc}_{sk}(s_{13}), s_{13}, s_{23}, \cdots, s_{(2n+2)3}) \end{bmatrix}$$

$$\rightarrow (\mathsf{Enc}_{sk}(r_{13}) \oplus \mathsf{Enc}_{sk}(s_{13}), r_{13} \oplus s_{13}, r_{23} \oplus s_{23}, \cdots, r_{(2n+2)3} \oplus s_{(2n+2)3})$$

Our fourth query Q_4 is a selective collection of the outputs in the previous round such that $Q_4 = (y_1 \oplus s_{12}, \cdots, y_n \oplus s_{n2}, r_{(n+1)2} \oplus s_{(n+1)2}, \cdots, r_{(2n)2} \oplus s_{(2n)2}, k \oplus r^*, \mathsf{Enc}_{sk}(r) \oplus \mathsf{Enc}_{sk}(s), r \oplus s)$. Notice that F and G can simulate all the computations of previous rounds upon Q_4. Hence, F and G can retrieve shared key *sk*. F computes $\mathsf{Enc}_{sk}(r_{13})$ and r_{13} by the simulation of computations on Q_3. Then, F checks to see if equality $\mathsf{Dec}_{sk}(\mathsf{Enc}_{sk}(r_{13}) \oplus (\mathsf{Enc}_{sk}(r_{13}) \oplus \mathsf{Enc}_{sk}(s_{13})))$ $= r_{13} \oplus (r_{13} \oplus s_{13})$ holds where $(\mathsf{Enc}_{sk}(r_{13}) \oplus \mathsf{Enc}_{sk}(s_{13}))$ and $(r_{13} \oplus s_{13})$ are obtained from Q_4. Since the equality holds, F deduces that the input query is indeed an adaptive query. Hence, F outputs $(k_\mathsf{F}, 0^n, 0^n, \cdots, 0^n)$ containing its secret key k_F. G does the same and outputs $(0^n, k_\mathsf{G}, 0^n, \cdots, 0^n)$. The individual outputs of F and G and the output of the parallel composition are described below.

$$Q_4 \rightarrow \begin{bmatrix} \mathsf{F} \rightarrow (k_\mathsf{F}, 0^n, 0^n, \cdots, 0^n) \\ \mathsf{G} \rightarrow (0^n, k_\mathsf{G}, 0^n, \cdots, 0^n) \end{bmatrix} \rightarrow (k_\mathsf{F}, k_\mathsf{G}, 0^n, \cdots, 0^n)$$

Now, it remains to prove that the above described functions are non-adaptively secure and their parallel composition is adaptively insecure. We prove the following claims that immediately substantiate Lemma 1. In this paper, a pseudo-random function is said to be *breakable by q adaptive queries* if there is a PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ distinguishes the pseudo-random function from a uniform random function by asking q adaptive queries to the pseudo-random function.

Claim. The function F and G described above are secure against any non-adaptive PPT adversary.

Claim. The parallel composition function $\mathsf{F} \oplus \mathsf{G}$ is breakable by four adaptive queries.

Lemma 6. *Suppose that a dense trapdoor permutation exists. Then, there exist non-adaptively secure pseudo-random functions* F *and* G *such that the parallel composition over XOR of* F *and* G *is breakable by four adaptive queries.*

3.2 Sequential Composition Insecurity from Dense Trapdoor Permutation

We now present a somewhat more interesting construction: namely a sequential composition of non-adaptively secure functions does not imply even *minimal* adaptive security.That is, we show that there exist non-adaptively secure pseudo-random functions F and G whose sequential composition is breakable by only two adaptive queries and yet it remains only non-adaptively secure.

Intuitions of Sequential Composition of F and G We provide the high-level overview of their formal constructions of counter-example PRFs F and G. The standard notions and specifications of the underlying primitives are identical to the ones in the previous section. F (resp. G) contains two secret keys k_{F} and $k_{\mathsf{F}}^{'}$ (resp. k_{G} and $k_{\mathsf{G}}^{'}$).

We define the first adaptive query Q_1 to be a fixed query, $(0^n, 0^n, \cdots, 0^n)$. Then, F and G are defined on Q_1 as follows.

- F computes (k, t_k) by $\mathsf{Gen}(1^n)$, a pair of a public key defining a one-way permutation and its corresponding trapdoor for the inverse permutation. F also computes pseudo-random strings $r_{21}, r_{31}, \cdots, r_{(2n+3)1}$ by $\mathsf{P}_{k_{\mathsf{F}}}(Q_1)$. F outputs $(k, r_{21}, \cdots, r_{(2n+3)1})$.
- On $(k, r_{21}, \cdots, r_{(2n+3)1})$, function G is defined to generate $2n+3$ pseudo-random strings $x_1, \ldots, x_n$, $s_{(n+1)1}, \cdots, s_{(2n+3)1}$ by $\mathsf{P}_{k_{\mathsf{G}}}(k, r_{21}, \cdots, r_{(2n+3)1})$ and computes the shared key $sk = b_1 b_2 \ldots b_i$, where $b_i = < x_i, s_{(n+i)1} >$ for all $1 \leq i \leq n$. In addition, G creates an encryption of $s_{(2n+1)1}$ with respect to the shared key, denoted by $\mathsf{Enc}_{sk}(s_{(2n+1)1})$. Also, G encrypts one of its own secrets $k_{\mathsf{G}}^{'}$ with respect to the shared key, resulting in $\mathsf{Enc}_{sk}(k_{\mathsf{G}}^{'})$. Finally, G encrypts x_is to y_i by a one-way permutation defined by k (i.e., $y_i = f_k(x_i)$ for all $1 \leq i \leq n$). Hence, G outputs $(y_1, \cdots, y_n, s_{(n+1)1}, \cdots, s_{(2n)1}$, $\mathsf{Enc}_{sk}(s_{(2n+1)1})$, $s_{(2n+1)1}$, $\mathsf{Enc}_{sk}(k_{\mathsf{G}}^{'}))$.

The computation of the sequential composition of F and G on Q_1 is described below:

$$\begin{aligned} Q_1 &\xrightarrow{\mathsf{F}} (k, r_{21}, \cdots, r_{(2n+3)1}) \\ &\xrightarrow{\mathsf{G}} (y_1, \cdots, y_n, s_{(n+1)1}, \cdots, s_{(2n)1}, \mathsf{Enc}_{sk}(s_{(2n+1)1}), s_{(2n+1)1}, \mathsf{Enc}_{sk}(k_{\mathsf{G}}^{'})) \end{aligned}$$

We define our second adaptive query Q_2 to be the output of the sequential composition on Q_1 such that $Q_2 = (y_1, \cdots, y_n, s_{(n+1)1}, \cdots, s_{(2n)1}, \mathsf{Enc}_{sk}(s_{(2n+1)1})$, $s_{(2n+1)1}, \mathsf{Enc}_{sk}(k_{\mathsf{G}}^{'}))$. On Q_2, we define F and G as follows.

- F obtains all x_is by inverting y_is with its private trapdoor information t_k as $f_{t_k}^{-1}(y_i)$ for all $1 \le i \le n$. Now F can retrieve the shared key sk by letting $sk = b_1 b_2 \cdots b_n$ where $b_i =< x_i, s_{(n+i)1} >$ for all $1 \le i \le n$. F takes $\mathsf{Enc}_{sk}(s_{(2n+1)1})$ from Q_2 and decrypts it to $s_{(2n+1)1}$ by $\mathsf{Dec}_{sk}(\mathsf{Enc}_{sk}(s_{(2n+1)1}))$. Finding the decrypted string equivalent to the $(2n+2)$th coordinate in Q_2 (i.e., $s_{(2n+1)1}$), F is convinced that Q_2 is an adaptive query. Then, F inverts the final coordinate of Q_2 with the shared key sk, so F obtains $k_{\mathsf{G}}^{'} = \mathsf{Dec}_{sk}(\mathsf{Enc}_{sk}(k_{\mathsf{G}}^{'}))$. Finally, F outputs a vector $(k_{\mathsf{G}}^{'},\ k_{\mathsf{F}},\ k_{\mathsf{F}}^{'},\ 0^n,\ \cdots,\ 0^n)$ containing all the secrets of F.
- Upon the input $(k_{\mathsf{G}}^{'}, k_{\mathsf{F}}, t_k, 0^n, \cdots, 0^n)$ from F, function G checks to see if the first coordinate of the input vector equals its own secret $k_{\mathsf{G}}^{'}$. Since the equality holds, G reveals all the secret keys of F and G by outputting $(k_{\mathsf{G}},\ k_{\mathsf{G}}^{'},\ k_{\mathsf{F}},\ k_{\mathsf{F}}^{'},\ 0^n,\ \cdots,\ 0^n)$.

All the individual outputs of F and G as a part of sequential composition is described as follows.

$$Q_2 \xrightarrow{\mathsf{F}} (k_{\mathsf{G}}, k_{\mathsf{F}}, k_{\mathsf{F}}^{'}, 0^n, \cdots, 0^n) \xrightarrow{\mathsf{G}} (k_{\mathsf{G}}, k_{\mathsf{G}}^{k'}, k_{\mathsf{F}}, k_{\mathsf{F}}^{'}, 0^n, \cdots, 0^n)$$

We prove the following claims that constitute Lemma 7 below. Hence, by Lemma 6 and Lemma 7, we immediately obtain Theorem 8.

Claim. The functions F and G described above are secure against any non-adaptive PPT adversary.

Claim. The sequential composition $\mathsf{G}(\mathsf{F}(\cdot))$ is breakable by two adaptive queries.

Lemma 7. *Suppose that a dense trapdoor permutation exists. Then, there exist non-adaptively secure functions* F *and* G *whose sequential composition* $\mathsf{G}(\mathsf{F}(\cdot))$ *is breakable by two adaptive queries.*

Theorem 8. *If a dense trapdoor permutation exists, then the composition of non-adaptively secure functions does not imply the adaptive security.*

4 Composition Insecurity vs. Uniform Transcript Key Agreement

In this section, we prove our main result: the existence of UTKA protocol implies the impossibility of obtaining adaptive security by the general composition of non-adaptively secure functions. Moreover, Pietrzak showed that the insecurity of sequential composition implies the existence of key agreement protocol. In fact, the key agreement protocol satisfies the property of *uniform-transcript* even though Pietrzak did not mention it in [10]. For the whole equality between the impossibility of general adaptively secure composition and UTKA, we prove that the parallel composition insecurity also achieves a UTKA by employing a small trick to the technique given in [10].

4.1 Parallel Composition Insecurity vs. Uniform Transcript Key Agreement

Constructing UTKA from the Adaptive Insecurity of $\mathsf{F} \oplus \mathsf{G}$ We present the parallel version of the result by using the technique originally presented by [10]. That is, if the parallel composition of two $k-1$ adaptively secure functions is not k-adaptively secure, then a $(2k-1)$-pass key agreement exists. For clarity, we rather present a special case where $k = 2$. Following the technique of [10], we construct a $(2k-1)$-pass bit agreement with ϵ-correlation and δ-security where ϵ is *non-negligible* and δ is *overwhelming*. It is known that n parallel repetitions of bit agreement with ϵ-correlation and δ-security achieves a n-bit key agreement without increasing the round complexity when ϵ is *noticeable* and δ is *overwhelming* [3]. With non-negligible ϵ, a bit agreement still realizes a key agreement which achieves correctness for (infinitely many) n such that for any c, $\epsilon \geq 1/n^c$.

We present the pictorial description of a $(2k-1)$-pass UTKA from two adaptively pseudo-random functions whose parallel composition is not k-adaptively secure when $k = 2$ in Protocol 1. The 3-pass uniform-transcript bit agreement

Protocol Bit-Agreement(1^n)

	Alice	Transcript	Bob	
b_{A}	$\$ \{0,1\}$			
k_{A}	$\mathsf{Gen}_{\mathsf{F}}(1^n)$		k_{B}	$\mathsf{Gen}_{\mathsf{G}}(1^n)$
x_1	$\mathcal{D}(1^n)$		x_1	$\mathcal{D}(1^n)$
If $b_{\mathsf{A}} = 0$,				
then z_1	$\mathsf{F}_{k_{\mathsf{A}}}(x_1)$			
else z_1	$\$ \{0,1\}^n$	$\xrightarrow{z_1}$		
		$\xleftarrow{y_1}$	y_1	$z_1 \oplus \mathsf{G}_{k_{\mathsf{B}}}(x_1)$
x_2	$\mathcal{D}(y_1)$		x_2	$\mathcal{D}(y_1)$
If $b_{\mathsf{A}} = 0$,				
then z_2	$\mathsf{F}_{k_{\mathsf{A}}}(x_2)$			
else z_2	$\$ \{0,1\}^n$	$\xrightarrow{z_2}$	y_2	$z_2 \oplus \mathsf{G}_{k_{\mathsf{B}}}(x_2)$
			b_{B}	$\mathcal{D}(y_1, y_2)$

Protocol 1: 3-pass uniform-transcript bit agreement based on 2-adaptive distinguisher $\mathcal{D}$

in Protocol 1 may be easily extended to the $(2k-1)$-pass bit agreement for arbitrary k.

Theorem 9. *Let* F *and* G *be k-adaptively secure pseudo-random functions. If the parallel composition* $\mathsf{F} \oplus \mathsf{G}$ *is NOT k-adaptively secure, then a $(2k-1)$-pass UTKA exists.*

Insecurity of Parallel Composition from UTKA A γ-round uniform-transcript key agreement protocol (γ-UTKA), denoted by $\mathsf{\Phi}_u^\gamma = (\mathsf{A}, \mathsf{B})$, is a uniform-transcript key agreement protocol consisting of two sub-protocols A and B, in which Alice (using A) and Bob (using B) exchange 2γ messages to each other (γ messages from each party) in order to share a secret key sk.

In this section, we use the parallel version of γ-UTKA to construct counter-example functions. The parallel γ-UTKA is a γ-UTKA where Alice and Bob are *symmetric* to each other in Protocol. In particular, Bob's first message is completely independent of Alice's first message and is only dependent on his own private randomness. That is, $\alpha_1 \quad \mathsf{A}_1(r_\mathsf{A})$ while $\beta_1 \quad \mathsf{B}_1(r_\mathsf{B})$ where r_A and r_B are independent randomness of Alice and Bob. For $2 \leq i \leq \gamma$, $\alpha_i \quad \mathsf{A}_i(r_\mathsf{A}, \beta_1, \cdots, \beta_{i-1})$ and $\beta_i \quad \mathsf{B}_i(r_\mathsf{B}, \alpha_1, \cdots, \alpha_{i-1})$. Finally, $sk \quad \mathsf{A}_{\gamma+1}(r_\mathsf{A}, \beta_1, \cdots, \beta_\gamma)$ and $sk \quad \mathsf{B}_{\gamma+1}(r_\mathsf{B}, \alpha_1, \cdots, \alpha_\gamma)$ where sk is the shared key.[3]

Now, we provide a high-level overview of our pseudo-random functions F and G from γ-UTKA and describe how to break the adaptive security of their parallel composition. For underlying primitives, we have a black-box access to $\Phi_u = (\mathsf{A}, \mathsf{B})$, parallel γ-UTKA described above. α_i and β_i denote the ith message computed by A and B respectively. We are given a pseudo-random private-key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ such that $\mathsf{Dec}_k(\mathsf{Enc}_k(x)) = x$. Finally, let P be any given adaptively secure PRP.

Intuitively, F utilizes A as its subroutine as well as G utilizes B as its subroutine in order for them to share a secret key via input and outputs. Then, F and G create pseudo-random strings specially related with respect to the shared secret key. As we input the specially related pseudo-random strings to the composition, the functions retrieve the shared key, verify the special relation hidden in the input query, and reveal their secret keys in their outputs. F and G internally contain secret keys k_F and k_G. F and G are defined over $(\{0,1\}^n)^{\gamma+2}$.

First, we define F and G upon the first adaptive (fixed) query $Q_1 = (0^n, \cdots, 0^n)$ as:

- F generates $\gamma + 2$ pseudo-random strings r_F, r_{21}, $\cdots$, $r_{(\gamma+2)1}$ by $\mathsf{P}_{k_\mathsf{F}}(Q_1)$. F creates Alice's first message α_1 by $\mathsf{A}_1(r_\mathsf{F})$ and then outputs (α_1, r_{21}, $\cdots$, $r_{(\gamma+2)1}$).
- G does the same as it generates s_G, s_{21}, $\cdots$, $s_{(\gamma+2)1}$ by $\mathsf{P}_{k_\mathsf{F}}(Q_1)$, and then computes Bob's first message β_1 by $\mathsf{B}_1(s_\mathsf{G})$, and outputs ($\beta_1$, s_{21}, $\cdots$, $s_{(\gamma+2)1}$).

[3] We emphasize that we can construct the same counter-example functions to show the same impossibility of adaptively secure composition by using a (*sequential*) $\gamma - UTKA$ in which Bob's first message is *dependent* on Alice's first message. However, it requires more adaptive queries to break the parallel composition of such functions. The main reason for using this parallel version of γ-UTKA is that it is simpler to emulate the key agreement protocol in the context of parallel composition of our proposed counter-example pseudo-random functions F and G. Also, it provides us with a tighter bound on the number of adaptive queries required to break the adaptive security of the parallel composition.

Below we depict the individual outputs of F and G on Q_1 and their parallel composition:

$$Q_1 \to \begin{bmatrix} \mathsf{F} \to (\alpha_1, r_{21}, \cdots, r_{(\gamma+2)1}) \\ \mathsf{G} \to (\beta_1, s_{21}, \cdots, s_{(\gamma+2)1}) \end{bmatrix} \to (\alpha_1 \oplus \beta_1, r_{21} \oplus s_{21}, \cdots, r_{(\gamma+2)1} \oplus s_{(\gamma+2)1})$$

Inductively, for $2 \le i \le \gamma$, we define F and G to process the i-th adaptive query $Q_i = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_{i-1} \oplus \beta_{i-1}, 0^n, \cdots, 0^n)$ as follows.

– F first regenerates r_{F} and α_1 by simulating the first-round computation. That is, F first computes $\mathsf{P}_{k_{\mathsf{F}}}(Q_1)$ to obtain r_{F} and then executes $\mathsf{A}(r_{\mathsf{F}})$. Then, F processes the following *chain of computations* in the direction of left-to-right and top-to-bottom with r_{F}, α_1 and Q_i,

$$\begin{array}{llll} \beta_1 & (\alpha_1 \oplus u_1) & \alpha_2 & \mathsf{A}_2(r_{\mathsf{F}}, \beta_1) \\ \vdots & & \vdots & \\ \beta_{i-1} & (\alpha_{i-1} \oplus u_{i-1}) & \alpha_i & \mathsf{A}_i(r_{\mathsf{F}}, \beta_1, \beta_2, \ldots, \beta_{i-1}) \end{array}$$

Finally, F outputs $(\alpha_i,\ r_{2i},\ \cdots,\ r_{(\gamma+2)i})$ where $r_{2i},\ \cdots,\ r_{(\gamma+2)i}$ are fresh pseudo-random strings generated by $\mathsf{P}_{k_{\mathsf{F}}}(Q_i)$.

– G is symmetrically defined. Hence, G outputs $(\beta_i,\ s_{2i},\ \cdots,\ s_{(\gamma+2)i})$ where $s_{2i},\ \cdots,\ s_{(\gamma+2)i}$ are pseudo-random strings generated by $\mathsf{P}_{k_{\mathsf{G}}}(Q_i)$.

On Q_i for $2 \le i \le \gamma$, we demonstrate the individual outputs of F and G and the output of their parallel composition below. Note that we obtain $\alpha_\gamma \oplus \beta_\gamma$ by feeding the parallel composition of F and G with Q_γ to be $(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_{\gamma-1} \oplus \beta_{\gamma-1}, 0^n, 0^n)$.

$$Q_i \to \begin{bmatrix} \mathsf{F} \to (\alpha_i, r_{2i}, \cdots, r_{(\gamma+2)i}) \\ \mathsf{G} \to (\beta_i, s_{2i}, \cdots, s_{(\gamma+2)i}) \end{bmatrix} \to (\alpha_i \oplus \beta_i, r_{2i} \oplus s_{2i}, \cdots, r_{(\gamma+2)i} \oplus s_{(\gamma+2)i})$$

The $(\gamma+1)$th adaptive query is defined to be $Q_{\gamma+1} = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, 0^n, 0^n)$. Then, we define our functions F and G on $Q_{\gamma+1}$ as follows.

– F first regenerates r_{F} and α_1 by simulating the first-round computation as before. Then, F performs the chain of computations described above, and so obtains $\beta_1, \beta_2, \cdots, \beta_\gamma$. Hence, F can generate a shared key sk by $\mathsf{A}_{\gamma+1}(r_{\mathsf{F}}, \beta_1, \beta_2, \ldots, \beta_\gamma)$. F generates pseudo-random strings $r_{1(\gamma+1)}$, $r_{2(\gamma+1)}$, $\cdots$, $r_{(\gamma+2)(\gamma+1)}$ by $\mathsf{P}_{k_{\mathsf{F}}}(Q_{\gamma+1})$. F creates an (pseudo-random) encryption $\mathsf{Enc}_{sk}(r_{1(\gamma+1)})$. F outputs $(\mathsf{Enc}_{sk}(r_{1(\gamma+1)}),\ r_{1(\gamma+1)},\ r_{3(\gamma+1)},\ \cdots,\ r_{(\gamma+2)(\gamma+1)})$.

– G is symmetrically defined. So, G outputs $(\mathsf{Enc}_{sk}(s_{1(\gamma+1)}),\ s_{1(\gamma+1)},\ s_{3(\gamma+1)},\ \cdots,\ s_{(\gamma+2)(\gamma+1)})$.

The following describes the each output of F and G, and that of parallel composition on $Q_{\gamma+1}$.

$$\begin{aligned} Q_{\gamma+1} \to & \begin{bmatrix} \mathsf{F} \to (\mathsf{Enc}_{sk}(r_{1(\gamma+1)}), r_{1(\gamma+1)}, r_{3(\gamma+1)}, \cdots, r_{(\gamma+2)(\gamma+1)}) \\ \mathsf{G} \to (\mathsf{Enc}_{sk}(s_{1(\gamma+1)}), s_{1(\gamma+1)}, s_{3(\gamma+1)}, \cdots, s_{(\gamma+2)(\gamma+1)}) \end{bmatrix} \\ & \to (\mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus \mathsf{Enc}_{sk}(s_{1(\gamma+1)}), r_{1(\gamma+1)} \oplus s_{1(\gamma+1)}, \\ & \qquad r_{3(\gamma+1)} \oplus s_{3(\gamma+1)}, \cdots, r_{(\gamma+2)(\gamma+1)} \oplus s_{(\gamma+2)(\gamma+1)}) \end{aligned}$$

The final $(\gamma+2)$th adaptive query is defined to be $Q_{\gamma+2} = (\alpha_1 \oplus \beta_1, \cdots, \alpha_\gamma \oplus \beta_\gamma, \mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus \mathsf{Enc}_{sk}(s_{1(\gamma+1)}), r_{1(\gamma+1)} \oplus s_{1(\gamma+1)})$ which is the combination of all the outputs of the parallel composition on the previous adaptive queries. Then, F and G are defined on $Q_{\gamma+2}$ as follows.

– F executes the chain of computations to retrieve $\beta_1, \beta_2, \cdots, \beta_\gamma$, then computes a shared key *sk* by $\mathsf{A}_{\gamma+1}(r_\mathsf{F}, \beta_1, \beta_2, \ldots, \beta_\gamma)$. Since $Q_{\gamma+1} = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, 0^n, 0^n)$, F can obtain $\mathsf{Enc}_{sk}(r_{1(\gamma+1)})$ and $r_{1(\gamma+1)}$ generated by the internal *simulation* of $\mathsf{F}(Q_{\gamma+1})$. F checks to see if equality $\mathsf{Dec}_{sk}(\mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus (\mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus \mathsf{Enc}_{sk}(s_{1(\gamma+1)}))) = r_{1(\gamma+1)} \oplus (r_{1(\gamma+1)} \oplus s_{1(\gamma+1)})$ holds where $(\mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus \mathsf{Enc}_{sk}(s_{1(\gamma+1)}))$ and $(r_{1(\gamma+1)} \oplus s_{1(\gamma+1)})$ are obtained from $Q_{\gamma+2}$. As the equality holds, F is convinced that $Q_{\gamma+2}$ is indeed an adaptively generated query. Hence, F outputs (k_F, 0^n, 0^n, $\cdots$, 0^n).
– G is symmetrically defined. Hence, G similarly outputs $(0^n, k_\mathsf{G}, 0^n, \cdots, 0^n)$.

Below we provide the overall picture of the individual computations of F and G and the output of their parallel composition.

$$Q_{\gamma+2} \rightarrow \begin{bmatrix} \mathsf{F} \rightarrow (k_\mathsf{F}, 0^n, 0^n, \cdots, 0^n) \\ \mathsf{G} \rightarrow (0^n, k_\mathsf{G}, 0^n, \cdots, 0^n) \end{bmatrix} \rightarrow (k_\mathsf{F}, k_\mathsf{G}, 0^n, \cdots, 0^n)$$

We prove the following claims that substantiate Theorem 10. Therefore, we immediately obtains Theorem 11 by Theorem 9 and 10.

Claim. The functions F and G described above are secure against any non-adaptive PPT adversary.

Claim. The parallel composition $\mathsf{F} \oplus \mathsf{G}$ is breakable by γ+2 adaptive queries.

Theorem 10. *If γ-UTKA $\Phi_u = (\mathsf{A}, \mathsf{B})$ exists, then there exist non-adaptively secure pseudo-random functions F and G such that their parallel composition over XOR is (γ+2)-adaptive query breakable.*

Theorem 11. *The parallel composition of two pseudo-random functions does not imply adaptive security if and only if the uniform-transcript key agreement exists.*

4.2 Sequential Composition Insecurity vs. Uniform Transcript Key Agreement

We examine the equivalence between the insecurity of sequential composition and the existence of UTKA protocol. Pietrzak already showed that a key agreement protocol can be achieved from two functions whose sequential composition is not adaptively secure. His key agreement protocol satisfies the property of uniform-transcript. We prove this as a separate claim in [1] and formally restate Pietrzak's theorem below.

Theorem 12 ([10]). *Let F and G be k-adaptively secure pseudo-random functions. If the sequential composition $\mathsf{G}(\mathsf{F}(\cdot))$ is NOT k-adaptively secure, then a (2k-1)-pass UTKA exists.*

Insecurity of Sequential Composition from UTKA In this section, we use the sequential version of γ-UTKA in which Bob's first message is *dependent* on Alice's first message to construct the counter-example PRFs. That is, β_1 $\mathsf{B}_1(r_\mathsf{B}, \alpha_1)$ where α_1 $\mathsf{A}_1(r_\mathsf{A})$ for r_A and r_B, independent randomness of Alice and Bob. For $2 \leq i \leq \gamma$, α_i $\mathsf{A}_i(r_\mathsf{A}, \beta_1, \cdots, \beta_{i-1})$ and β_i $\mathsf{B}_i(r_\mathsf{B}, \alpha_1, \cdots, \alpha_i)$. Consequently, sk $\mathsf{A}_{\gamma+1}(r_\mathsf{A}, \beta_1, \cdots, \beta_\gamma)$ and sk $\mathsf{B}_{\gamma+1}(r_\mathsf{B}, \alpha_1, \cdots, \alpha_\gamma)$ where sk is the shared key. Notice that in this scenario Bob must wait for the first message α_1 from Alice in order to compute his first message β_1.

In the following, we present the high-level overview on our constructions of counter-example functions F and G based on γ-UTKA described above. For the building blocks, we are given a sequential version of γ-UTKA, $\Phi_u = (\mathsf{A}, \mathsf{B})$ and all the other primitives remain identical to the ones in Section 4.1. F (resp. G) is defined over $(\{0,1\}^n)^{\gamma+3}$ and internally possesses a secret key k_F (resp. k_G).

Our first adaptive query is an arbitrary vector in $(\{0,1\}^n)^{\gamma+3}$ as $Q_1 = (u_1, u_2, \cdots, u_{\gamma+2}, u^*)$ for $u_1, u_2, \cdots, u_{\gamma+2}, u^*$ $_\$$ $\{0,1\}^n$. On Q_1, we define F and G as follows.

– F computes a pseudo-random string r_F by $\mathsf{P}_{k_\mathsf{F}}(u^*)$. Then, F generates the first message α_1 by executing $\mathsf{A}_1(r_\mathsf{F})$. F continues to compute $r_{21}, \cdots, r_{\gamma 1}$ by executing $\mathsf{A}_2(r_\mathsf{F}, u_1), \cdots, \mathsf{A}_\gamma(r_\mathsf{F}, u_1, \cdots, u_{\gamma-1})$. Notice that Q_1 is an arbitrarily chosen input so that running A (Alice) on Q_1 produces only pseudo-random strings except for the first message α_1. F computes its first n-bit shared key sk^1_F from $\mathsf{A}_{\gamma+1}(r_\mathsf{F}, u_1, \cdots, u_\gamma)$. F tests if $\mathsf{Dec}_{sk^1_\mathsf{F}}(u_{\gamma+1}) = u_{\gamma+2}$. The equality is satisfied only negligible probability since $u_{\gamma+1}$ and $u_{\gamma+2}$ are arbitrary chosen. Hence, with overwhelming probability, F concludes its computation by outputting $(\alpha_1, r_{21}, r_{31}, \cdots, \mathsf{Enc}_{sk^1_\mathsf{F}}(r_{(\gamma+1)1}), r_{(\gamma+1)1}, r_{(\gamma+3)1})$ where $r_{(\gamma+1)1}$, $r_{(\gamma+2)1}$ and $r_{(\gamma+3)1}$ are generated from $\mathsf{P}_{k_\mathsf{F}}(u_{\gamma+1}, u_{\gamma+2}, u_{\gamma+3})$.
– On $\mathsf{F}(Q_1)$, G is defined to compute β_1 by $\mathsf{B}_1(s_\mathsf{G}, \alpha_1)$ where s_G is generated by $\mathsf{P}_{k_\mathsf{G}}(u_1)$ and α_1 is the first message validly generated by F. G continues to compute $s_{21}, \cdots, s_{\gamma 1}$ by executing $\mathsf{B}_2(s_\mathsf{G}, \alpha_1, r_{21}), \cdots, \mathsf{B}_\gamma(s_\mathsf{G}, \alpha_1, r_{21}, \cdots, r_{\gamma 1})$. Since $r_{21}, \cdots, r_{\gamma 1}$ are pseudo-random strings computed by F upon non-adaptive query Q_1, $s_{21}, \cdots, s_{\gamma 1}$ are pseudo-random strings. G computes sk^1_G from $\mathsf{B}_{\gamma+1}(r_\mathsf{G}, u_1, \cdots, u_\gamma)$ and then tests if $\mathsf{Dec}_{sk^1_\mathsf{G}}(u_{\gamma+1}) = u_{\gamma+2}$ holds. This equality holds with only negligible probability. G computes pseudo-random strings $s_{(\gamma+1)1}$, $s_{(\gamma+2)1}$ and $s_{(\gamma+3)1}$ from $\mathsf{P}_{k_\mathsf{G}}(\pi_{sk^1_\mathsf{F}}(r_{(\gamma+1)1}), r_{(\gamma+1)1}, r_{(\gamma+3)1})$. G outputs $(\beta_1, s_{21}, s_{31}, \cdots, \mathsf{Enc}_{sk^1_\mathsf{G}}(s_{(\gamma+1)1}), s_{(\gamma+1)1}, s_{(\gamma+3)1})$.

We describe the outputs of F and G in the computation of their sequential composition on Q_1:

$$\begin{aligned} Q_1 &\xrightarrow{\mathsf{F}} (\alpha_1, r_{21}, r_{31}, \cdots, \mathsf{Enc}_{sk^1_\mathsf{F}}(r_{(\gamma+1)1}), r_{(\gamma+1)1}, r_{(\gamma+3)1}) \\ &\xrightarrow{\mathsf{G}} (\beta_1, s_{21}, s_{31}, \cdots, \mathsf{Enc}_{sk^1_\mathsf{G}}(s_{(\gamma+1)1}), s_{(\gamma+1)1}, s_{(\gamma+3)1}). \end{aligned}$$

Inductively, for $2 \leq i \leq \gamma - 1$, the ith adaptive query Q_i is in the form of $(\beta_1, \cdots, \beta_{i-1}, s_{i(i-1)}, \cdots, s_{\gamma(i-1)}, \mathsf{Enc}_{sk^{i-1}_\mathsf{G}}(s_{(\gamma+1)(i-1)}), s_{(\gamma+1)(i-1)}, u^*)$ where u^* is the final coordinate of Q_1 and the rest of coordinates are the first $2\gamma +$

2 coordinates in the output of $\mathsf{G}(\mathsf{F}(Q_{i-1}))$. Then, F computes all the messages α_1 to α_γ and shared key sk_{F}^i based on Q_i as described above. F tests if $\mathsf{Dec}_{sk_{\mathsf{F}}^i}(\mathsf{Enc}_{sk_{\mathsf{G}}^{i-1}}(s_{(\gamma+1)(i-1)})) = s_{(\gamma+1)(i-1)}$. Obviously, $sk_{\mathsf{F}}^i \neq sk_{\mathsf{G}}^{i-1}$ with overwhelming probability since the keys are computed based on insufficient number of valid messages. Hence, F outputs (α_1, $\cdots$, α_i, $r_{(i+1)i}$, $\cdots$, $r_{(\gamma)i}$, $\mathsf{Enc}_{sk_{\mathsf{F}}^i}(r_{(\gamma+1)i})$, $r_{(\gamma+1)i}$, $r_{(\gamma+3)i}$). Similarly, G undertakes the same course of computations: G computes messages and shared key, tests the equality and finally outputs (β_1, $\cdots$, β_i, $s_{(i+1)i}, \cdots, s_{(\gamma)i}, \mathsf{Enc}_{sk_{\mathsf{G}}^i}(s_{(\gamma+1)i}), s_{(\gamma+1)i}, s_{(\gamma+3)i}$). The individual output of F and the output of G in their sequential composition on Q_i are described as follows:

$$
\begin{aligned}
Q_i &\xrightarrow{\mathsf{F}} (\alpha_1, \cdots, \alpha_i, r_{(i+1)i}, \cdots, r_{(\gamma)i}, \mathsf{Enc}_{sk_{\mathsf{F}}^i}(r_{(\gamma+1)i}), r_{(\gamma+1)i}, r_{(\gamma+3)i}) \\
&\xrightarrow{\mathsf{G}} (\beta_1, \cdots, \beta_i, s_{(i+1)i}, \cdots, s_{(\gamma)i}, \mathsf{Enc}_{sk_{\mathsf{G}}^i}(s_{(\gamma+1)i}), s_{(\gamma+1)i}, s_{(\gamma+3)i}).
\end{aligned}
$$

Hence, after the $(\gamma-1)$th adaptive query, our γth adaptive query Q_γ is (β_1, β_2, $\cdots$, $\beta_{\gamma-1}$, $s_{\gamma(\gamma-1)}$, $\mathsf{Enc}_{sk_{\mathsf{G}}^{\gamma-1}}(s_{(\gamma+1)(\gamma-1)})$, $s_{(\gamma+1)(\gamma-1)}$, u^*). On Q_γ, we define F and G as follows.

- F computes r_{F} from $\mathsf{P}_{k_{\mathsf{F}}}(u^*)$. Then, F internally regenerates all α_i by $\mathsf{A}_i(r_{\mathsf{F}}$, $\beta_1, \cdots, \beta_{i-1})$ for $1 \leq i \leq \gamma$ and shared key sk_{F}^γ by $\mathsf{A}_i(r_{\mathsf{F}}, \beta_1, \cdots, \beta_{i-1}, s_{\gamma(\gamma-1)})$. sk_{F}^γ is still a merely pseudo-random string since $s_{\gamma(\gamma-1)}$ is not a proper message. F performs the equality test $\mathsf{Dec}_{sk_{\mathsf{F}}^\gamma}(\mathsf{Enc}_{sk_{\mathsf{G}}^{\gamma-1}}(s_{(\gamma+1)(\gamma-1)})) = s_{(\gamma+1)(\gamma-1)}$ which fails with overwhelming probability. Hence, F outputs (α_1, $\cdots$, α_γ, $\mathsf{Enc}_{sk_{\mathsf{F}}^\gamma}(r_{(\gamma+1)\gamma})$, $r_{(\gamma+1)\gamma}$, $r_{(\gamma+3)\gamma}$) as ($r_{(\gamma+1)\gamma}, r_{(\gamma+2)\gamma}$, $r_{(\gamma+3)\gamma}$) is generated by $\mathsf{P}_{k_{\mathsf{F}}}(\mathsf{Enc}_{sk_{\mathsf{G}}^{\gamma-1}}(s_{(\gamma+1)(\gamma-1)})$, $s_{(\gamma+1)(\gamma-1)}$, u^*).
- G obtains r_{G} by $\mathsf{P}_{k_{\mathsf{G}}}(\alpha_1)$. Then, since G obtains its complete set of γ messages α_i's from F, function G correctly generates all the messages β_i's by executing $\mathsf{B}_i(r_{\mathsf{G}}, \alpha_1, \cdots, \alpha_i)$ for all $1 \leq i \leq \gamma$. In addition, G computes the shared key sk_{G}^γ from executing $\mathsf{B}_{\gamma+1}(r_{\mathsf{G}}, \alpha_1, \cdots, \alpha_\gamma)$. Finally, G outputs $(\beta_1, \cdots, \beta_\gamma, \mathsf{Enc}_{sk_{\mathsf{G}}^\gamma}(s_{(\gamma+1)\gamma}), s_{(\gamma+1)\gamma}, s_{(\gamma+3)\gamma})$ since $\mathsf{Dec}_{sk_{\mathsf{G}}^\gamma}(\mathsf{Enc}_{sk_{\mathsf{F}}^\gamma}(r_{(\gamma+1)(\gamma)}))$ $\neq r_{(\gamma+1)(\gamma)}$ with overwhelming probability, where ($s_{(\gamma+1)\gamma}, s_{(\gamma+2)\gamma}$, $s_{(\gamma+3)\gamma}$) is generated by $\mathsf{P}_{k_{\mathsf{G}}}(\mathsf{Enc}_{sk_{\mathsf{F}}^\gamma}(r_{(\gamma+1)\gamma})$, $r_{(\gamma+1)\gamma}$, $r_{(\gamma+3)\gamma}$).

We describe the overall picture of F and G in their sequential composition on input Q_γ below:

$$
\begin{aligned}
Q_\gamma &\xrightarrow{\mathsf{F}} (\alpha_1, \cdots, \alpha_\gamma, \mathsf{Enc}_{sk_{\mathsf{F}}^\gamma}(r_{(\gamma+1)\gamma}), r_{(\gamma+1)\gamma}, r_{(\gamma+3)\gamma}) \\
&\xrightarrow{\mathsf{G}} (\beta_1, \cdots, \beta_\gamma, \mathsf{Enc}_{sk_{\mathsf{G}}^\gamma}(s_{(\gamma+1)\gamma}), s_{(\gamma+1)\gamma}, s_{(\gamma+3)\gamma}).
\end{aligned}
$$

The (final) $(\gamma+1)$th adaptive query $Q_{\gamma+1}$ is defined to be ($\beta_1, \cdots, \beta_\gamma$, $\mathsf{Enc}_{sk_{\mathsf{G}}^\gamma}(s_{(\gamma+1)\gamma})$, $s_{(\gamma+1)\gamma}, u^*$). On $Q_{\gamma+1}$, we define functions F and G on $Q_{\gamma+1}$ as:

- F now obtains all the messages β_i's from $Q_{\gamma+1}$ so that it can compute all the messages $\alpha_1, \cdots, \alpha_\gamma$ and the shared key $sk_{\mathsf{F}}^{\gamma+1}$ by executing $\mathsf{A}_{\gamma+1}(r_{\mathsf{F}}, \beta_1, \cdots, \beta_\gamma)$. F tests if the following equality is satisfied: $\mathsf{Dec}_{sk_{\mathsf{F}}^{\gamma+1}}(\mathsf{Enc}_{sk_{\mathsf{G}}^\gamma}(s_{(\gamma+1)(\gamma)})) =$

$s_{(\gamma+1)(\gamma)}$. Notice that $sk_{\mathsf{F}}^{\gamma+1} = sk_{\mathsf{G}}^{\gamma}$ since both keys are computed on each complete set of messages. Hence, F verifies that the equality holds and is convinced that $Q_{\gamma+1}$ is adaptively generated. Finally, F outputs $(\alpha_1, \cdots, \alpha_\gamma, \pi_{sk_{\mathsf{F}}^{\gamma+1}}(r_{(\gamma+1)(\gamma+1)}), r_{(\gamma+1)(\gamma+1)}, k_{\mathsf{F}})$ where $r_{(\gamma+1)(\gamma+1)}, r_{(\gamma+2)(\gamma+1)}$ and $r_{(\gamma+3)(\gamma+1)}$ are generated from $\mathsf{P}_{k_{\mathsf{F}}}(\mathsf{Enc}_{sk_{\mathsf{G}}^{\gamma}}(s_{(\gamma+1)\gamma}), s_{(\gamma+1)\gamma}, u^*)$.

– On $(\alpha_1, \cdots, \alpha_\gamma, \pi_{sk_{\mathsf{F}}^{\gamma+1}}(r_{(\gamma+1)(\gamma+1)}), r_{(\gamma+1)(\gamma+1)}, k_{\mathsf{F}})$, G also computes all of the messages and shared key $sk_{\mathsf{G}}^{\gamma+1}$. Clearly, $sk_{\mathsf{F}}^{\gamma+1} = sk_{\mathsf{G}}^{\gamma+1}$ since both keys are computed based on the same set of messages $\alpha_1 \cdots \alpha_\gamma$. Then G tests if $\mathsf{Dec}_{sk_{\mathsf{G}}^{\gamma+1}}(\mathsf{Enc}_{sk_{\mathsf{F}}^{\gamma+1}}(r_{(\gamma+1)(\gamma+1)})) = r_{(\gamma+1)(\gamma+1)}$. Since both $sk_{\mathsf{F}}^{\gamma+1}$ and $sk_{\mathsf{G}}^{\gamma+1}$ are computed from the complete sets of messages, they must be equal. G is convinced that the query from F is adaptively generated. Therefore, G outputs $(k_{\mathsf{G}}, k_{\mathsf{F}}, 0^n, \cdots, 0^n)$ where k_{F} can be obtained from the input (i.e., the final coordinate of the input vector).

The overall description of outputs of F and G on the final adaptive query is provided below:

$$Q_{\gamma+1} \xrightarrow{\mathsf{F}} (\alpha_1, \cdots, \alpha_\gamma, \mathsf{Enc}_{sk_{\mathsf{F}}^{\gamma+1}}(r_{(\gamma+1)(\gamma+1)}), r_{(\gamma+1)(\gamma+1)}, k_{\mathsf{F}}) \xrightarrow{\mathsf{G}} (k_{\mathsf{G}}, k_{\mathsf{F}}, 0^n, \cdots, 0^n).$$

We prove the following claims which substantiate Theorem 13. Putting Theorem 12 and 13 together, we immediately obtains Theorem 14.

Claim. The functions F and G described above are secure against any non-adaptive PPT adversary.

Claim. The sequential composition of functions F and G, defined by $\mathsf{S}(\cdot) = \mathsf{G}(\mathsf{F}(\cdot))$, is breakable by $\gamma + 1$ adaptive queries.

Theorem 13. *If γ-UTKA $\Phi_u = (\mathsf{A}, \mathsf{B})$ exists, then there exist non-adaptively secure functions F and G such that the sequential composition $\mathsf{G}(\mathsf{F}(\cdot))$ is $(\gamma+1)$-adaptive query breakable.*

Theorem 14. *The sequential composition of two pseudo-random functions does not imply adaptive security if and only if the uniform-transcript key agreement exists.*

5 Impossibility of Adaptively Secure Self-Composition

Self-composition is a composition of two or more copies of a single function. For instance, we call $\mathsf{F}(\mathsf{F}(\cdot))$ the sequential self-composition of function F, and $\mathsf{F} \oplus \mathsf{F}$ the parallel self-composition of function F. Note that several copies of identical F's must contain independent secret seeds. That is, each copy of F's must be allowed to be independently drawn from its function family.

So far, we proved the equivalence relation between the insecurity of composition and UTKA protocols. In fact, when we mention the insecurity of composition in previous sections, the main argument is rather that, given a non-adaptively secure function, there might be another non-adaptively secure function such that their composition is adaptively insecure. We call this type of composition *general-composition*. Hence, we still have a lingering unanswered question of whether the self-composition of a non-adaptively secure function implies the unconditional adaptive security. We answered the question negatively as follows.

Suppose that we are given non-adaptively secure pseudo-random functions F_k and $\mathsf{G}_{k'}$, without loss of generality, both defined over $\{0,1\}^n$ such that their parallel (general-)composition $(\mathsf{F} \oplus \mathsf{G})(\cdot)$ is adaptively insecure. Note that k and k' are independently chosen secret seeds for pseudo-random functions. That is, there exists a PPT adversary $\mathcal{A}$ with an adaptive adversarial strategy which succeeds in breaking the security of $(\mathsf{F} \oplus \mathsf{G})(\cdot)$ with non-negligible probability δ. Now, we define a function family $\mathcal{F}_{(b,s)} : \{0,1\}^n \to \{0,1\}^n$ on input string u by

$$\mathcal{F}_{(b,s)}(u) = \begin{cases} \mathsf{F}_s(u) & \text{if b} = 0 \\ \mathsf{G}_s(u) & \text{if b} = 1 \end{cases} \tag{$*$}$$

where b and s are private seeds.

It is easy to see that function $\mathcal{F}(\cdot)$ is also non-adaptively secure due to the non-adaptive security of functions F and G. This trivially leads to

$$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{F}} \leq \mathbf{Adv}_{\mathcal{A}}^{\mathsf{F}} + \mathbf{Adv}_{\mathcal{A}}^{\mathsf{G}}.$$

To break the adaptive security of $(\mathcal{F} \oplus \mathcal{F})(\cdot)$, it suffices to draw two copies of functions from the family at random and then use the same adaptively adversarial strategy of $\mathcal{A}$ as follows: the first bit of seeds of F and G differ in their first bit with probability $1/2$. Therefore, if we draw two independent $\mathcal{F}$'s, then $\mathcal{F} \oplus \mathcal{F}$ is equivalent to $\mathsf{F} \oplus \mathsf{G}$ with probability $1/4$ which is adaptively insecure.

Informally, by the above construction of $\mathcal{F}$ from any two non-adaptively secure functions F and G such that their parallel composition is not adaptively secure, we actually show that the adaptive insecurity of the parallel general-composition implies the adaptive insecurity of the parallel self-composition. We formally state this as follows.

Theorem 15. *Suppose there are two non-adaptively secure functions F and G such that the parallel composition $(\mathsf{F} \oplus \mathsf{G})(\cdot)$ is adaptively insecure. Then, there exists a non-adaptively secure function $\mathcal{F}$ such that the parallel self-composition is adaptively insecure.*

Combining the above theorem with the previous results of this paper in Sections 3.1 and 4.1 related to parallel composition insecurity from DTP and γ-UTKA, we obtain the following theorems.

Theorem 16. *If a family of dense trapdoor permutations or a UTKA exists, then the parallel self-composition of a non-adaptively secure function does not imply adaptive security.*

Furthermore, the above constructions of $\mathcal{F}$ defined in $(*)$ and its analysis of adaptive security can be easily extended to the context of sequential composition.

In particular, $\mathcal{F}$ is also non-adaptively secure while $\mathcal{F}(\mathcal{F}(\cdot))$ is equal to $\mathsf{G}(\mathsf{F}(\cdot))$ with probability $1/4$ when we draw two independent $\mathcal{F}$'s from its function family. Thus, $\mathcal{F}(\mathcal{F}(\cdot))$ is also adaptively insecure. Consequently, we obtain the following theorem.

Theorem 17. *Suppose there are two non-adaptively secure functions* F *and* G *such that the sequential composition* $\mathsf{G}(\mathsf{F}(\cdot))$ *is adaptively insecure. Then, there exists a non-adaptively secure function* $\mathcal{F}$ *such that the self-composition is adaptively insecure.*

Again, combining the above theorem with the previous results of this paper in Sections 3.2 and 4.2 relevant to sequential composition insecurity from DTP and γ-UTKA, we derive the following theorem.

Theorem 18. *If a family of dense trapdoor permutations or a UTKA exists, then the sequential self-composition of a non-adaptively secure function does not imply adaptive security.*

References

1. Cho, C., Lee, C.K., Ostrovsky, R.: Equivalence of uniform key agreement and composition insecurity. Electronic Colloquium on Computational Complexity (ECCC), Report No. 108 (2009)
2. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM 33(4), 792–807 (1986)
3. Holenstein, T.: Key agreement from weak bit agreement. In: STOC '05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing. pp. 664–673. ACM, New York, NY, USA (2005)
4. Impagliazzo, R.: A personal view of average-case complexity. In: SCT '95: Proceedings of the 10th Annual Structure in Complexity Theory Conference. p. 134. IEEE Computer Society, Washington, DC, USA (1995)
5. Luby, M., Rackoff, C.: Pseudo-random permutation generators and cryptographic composition. In: STOC '86: Proceedings of the 18th Annual ACM Symposium on Theory of Computing. pp. 356–363. ACM, New York, NY, USA (1986)
6. Maurer, U., Pietrzak, K.: Composition of random systems: When two weak make one strong. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 410–427. Springer, Heidelberg (2004)
7. Maurer, U., Pietrzak, K., Renner, R.: Indistinguishability amplification. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 130–149. Springer, Heidelberg (2007)
8. Myers, S.: Black-box composition does not imply adaptive security. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 189–206. Springer, Heidelberg (2004)
9. Pietrzak, K.: Composition does not imply adaptive security. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 55–65. Springer, Heidelberg (2005)
10. Pietrzak, K.: Composition implies adaptive security in minicrypt. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 328–338. Springer, Heidelberg (2006)
11. Vaudenay, S.: Decorrelation: A theory for block cipher security. J. Cryptology 16(4), 249–286 (2003)