Electronic Colloquium on Computational Complexity, Report No. 108 (2009)

Equivalence of Uniform Key Agreement and Composition Insecurity

Chongwon Cho* Chen-Kuei Lee† Rafail Ostrovsky‡

University of California, Los Angeles
Computer Science Department
Los Angeles, CA, 90095-1596, USA

{ccho, jcklee, rafail}@cs.ucla.edu

October 31, 2009

Abstract

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.

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

*Supported in part by NSF grants 0716835, 0716389, 0830803, 0916574

†Supported in part by NSF grants 0716835, 0716389, 0830803, 0916574

‡Supported in part by IBM Faculty Award, Xerox Innovation Group Award, OKAWA Research Award, NSF grants 0430254, 0716835, 0716389, 0830803, 0916574, BSF grant, and U.C. MICRO grant

ISSN 1433-8092

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 composition rather belongs to the world of private-key cryptography (also known as "minicrypt"). For further discussion on cryptomania and minicrypt, see [Imp95].

Now, let us briefly recall the definition of Pseudo-Random Functions (PRF) [GGM86]. 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 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 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 [Vau03] 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 [LR86] show the similar security amplification result in the information theoretical context.

In the information theoretic setting, Maurer and Pietrzak [MP04] proved that composition of non-adaptive secure functions amplifies its security ϵ to security $2\epsilon(1+\ln(\epsilon^{-1}))$ against an adaptive distinguisher.

Myers [Mye04] showed that the existence of oracles relative to which there are non-adaptively secure permutations, but where the composition of such generators fails to achieve adaptive security.

Recently, Pietrzak [Pie05] 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 [Pie06] implies a slightly stronger result: that his key agreement protocol satisfies the property of uniform-transcript (we show this fact in the appendix). Thus, we can restate the Pietrazak's result as follows:

Theorem 1. [Pie06] *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 adaptively security.*

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

Theorem 4. *If a parallel composition of pseudo-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 (parallel or sequential) 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.*

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 Figure 1.

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 [Pie05], 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 discuss the implications of out main results on the impossibility of adaptively secure self-composition.

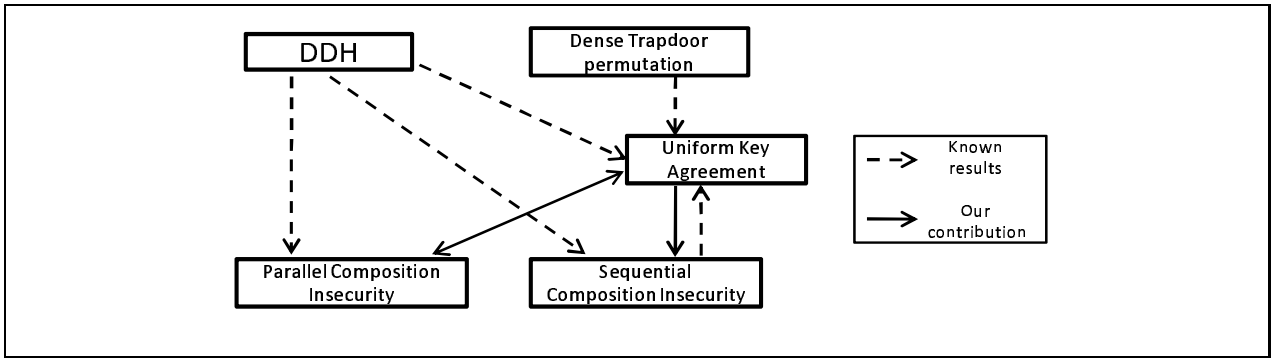

Figure 1: Relationship between composition insecurity and other assumptions

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 $x \leftarrow_{\$} \{0,1\}^n$ when string x is uniformly drawn from $\{0,1\}^n$. For further discussion on the following definitions and notions, see [Gol01].

Definition 1 (Negligible and Overwhelming)**.** *A function $\epsilon : \mathbb{N} \to \mathbb{R}$ is **negligible** if for every $c > 0$ there exists an N_c such that for all $n > N_c$, $\epsilon(n) \leq 1/n^c$. On the other hand, $1 - \epsilon$ is said to be **overwhelming** in n.*

Definition 2 (Non-negligible)**.** *A function $\delta : \mathbb{N} \to \mathbb{R}$ is **non-negligible** if for every $c > 0$ there exists infinitely many n such that $\delta(n) \geq 1/n^c$.*

Definition 3 (Noticeable)**.** *A function $\mu : \mathbb{N} \to \mathbb{R}$ is **noticeable** if for every $c > 0$ there exists an N_c such that for all $n > N_c$, $\mu(n) \geq 1/n^c$.*

Definition 4 (Polynomial Indistinguishability)**.** *Two probability ensembles $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are **polynomially indistinguishable** if for every Probabilistic Polynomial-Time (PPT) algorithm (distinguisher) $\mathcal{A}$, there exists a negligible function ϵ such that: for all random coin tosses r and r' of $\mathcal{A}$,*

$$|\Pr[\mathcal{A}_r(X_n) = 1] - \Pr[\mathcal{A}_{r'}(Y_n) = 1]| \leq \epsilon(n).$$

Definition 5 (Pseudo-Random Function (Permutation))**.** *Given a randomly chosen key $k \in K$ for a key space K, an efficiently computable keyed function $F_k : K \times \{0,1\}^n \to \{0,1\}^n$ is called pseudo-random function (PRF), if for every probabilistic polynomial-time algorithm (distinguisher) $\mathcal{A}$, given access to the function F_k and a uniform random function $U : \{0,1\}^n \to \{0,1\}^n$ and for all n, there exists a negligible function $\epsilon(n)$ such that: for all random coin tosses r and r' of $\mathcal{A}$,*

$$\left|\Pr[\mathcal{A}_r^{F_k}(1^n) = 1] - \Pr[\mathcal{A}_{r'}^{U}(1^n) = 1]\right| \leq \epsilon(n).$$

In addition, if F is one-to-one and onto for all k, then we call F_k a pseudo-random permutation (PRP).

Definition 6 (Distinguishing Advantage)**.** *Given a polynomial-time distinguisher $\mathcal{A}(q,t)$, which runs in time t and makes at most q queries to the oracle O (i.e., a pseudo-random function F) or random function R, we define the advantage of $\mathcal{A}(q,t)$ as: for all random coin tosses r and r' of $\mathcal{A}$,*

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{O}}(q,t) = \max_{\mathcal{A}} \left|\Pr[\mathcal{A}_r^{\mathsf{O}} = 1] - \Pr[\mathcal{A}_{r'}^{\mathsf{R}} = 1]\right|.$$

A distinguisher is a *non-adaptive* distinguisher if it makes all the queries before it receives the output. Otherwise, we call it an *adaptive* distinguisher.

Definition 7 (Non-Adaptive Versus Adaptive Security)**.** *A pseudo-random function f is non-adaptively secure if, for all polynomial-time non-adaptive distinguisher $\mathcal{A}(q,t)$, there exists a negligible function ϵ such that $\mathbf{Adv}^{\mathsf{O}}_{\mathcal{A}}(q,t) \leq \epsilon$. On the contrary, we say a pseudo-random function f is adaptively secure if, for all polynomial-time adaptive distinguisher $\mathcal{A}(q,t)$, there exists a negligible function ϵ such that $\mathbf{Adv}^{\mathsf{O}}_{\mathcal{A}}(q,t) \leq \epsilon$.*

Definition 8 (Parallel and Sequential Composition)**.** *The parallel XOR-composition of two functions F and G, denoted as $\mathsf{F} \oplus \mathsf{G}$, is the operation that composes the output value of F and G over the bit-wise Exclusive-Or (XOR) operation. The sequential composition of F and G, denoted as $\mathsf{F} \circ \mathsf{G}$, is the operation that applies two functions sequentially, i.e., $\mathsf{F} \circ \mathsf{G}(\cdot) = \mathsf{G}(\mathsf{F}(\cdot))$.*

Informally, a dense trapdoor permutation family is a trapdoor permutation family with a polynomially dense public description of permutation so that a public description is indistinguishable from uniform random [SP92, Hai04].

Definition 9 (Dense Trapdoor One-Way Permutation (DTP))**.** *The algorithm triplet (Gen, f_k, $f_{t_k}^{-1}$) is a family of dense trapdoor permutations if the following hold:*

- $\mathsf{Gen}(1^n)$ *is a probabilistic polynomial-time algorithm such that on input 1^n, it outputs a pair of two strings $k \in \{0,1\}^n$ and t_k, where $|t_k| \leq p(n)$.*
- *Given k, algorithm $f_k : \{0,1\}^n \to \{0,1\}^n$ is a polynomial-time computable permutation.*
- *Given t_k, algorithm $f_{t_k}^{-1}$ is a polynomial-time computable inverse permutation of f_k. That is, $f_{t_k}^{-1}(f_k(x)) = x$ is efficiently computable for all $x \in \{0,1\}^n$.*
- *For all probabilistic polynomial-time algorithm $\mathcal{A}$, the following holds for any (k, t_k), $x \in \{0,1\}^n$, for all random coin toss r of $\mathcal{A}$,*

$$\Pr[\mathcal{A}_r^{f_k}(k, f_k(x)) = f_{t_k}^{-1}(f_k(x))] \leq \epsilon(n)$$

  where $\epsilon(n)$ is a negligible function in n.

- *For all probabilistic polynomial-time algorithm $\mathcal{A}$, there exists a negligible function ϵ such that, given access to function $\mathsf{Gen}(1^n)$ or uniform function U as oracle: for all random coin tosses r and r' of $\mathcal{A}$,*

$$\left|\Pr[\mathcal{A}_r^{\mathsf{Gen}}(1^n) = 1] - \Pr[\mathcal{A}_{r'}^{U}(1^n) = 1]\right| \leq \epsilon(n).$$

Definition 10 (Hard-Core Predicate)**.** *A polynomial-time computable function family $\mathsf{B} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a hard-core predicate of one-way function f if, for every probabilistic polynomial-time algorithm $\mathcal{A}$ and for all $x \in \{0,1\}^n$, there exists a negligible function ϵ such that*

$$\Pr[\mathcal{A}(f(x)) = \mathsf{B}(x,r))] \leq \frac{1}{2} + \epsilon(n).$$

Goldreich and Levin [GL89] presented the simple construction of hardcore predicate $\mathsf{B}(x,r)$ from any one-way function f as $b = \bigoplus_i x_i r_i$, as usual, denoted as $b = < x, r >$, the inner product of two vectors.

Definition 11 (Bit Agreement (BA)). *Informally, a bit agreement is a protocol in which two parties, Alice and Bob, secretly agree on a bit at the end. Formally, upon the security parameter n as a common input, Alice and Bob output a bit b_A and b_B, respectively. Then, the protocol is said to have the correlation $\epsilon(n)$ if for all n,*

$$\Pr[b_A = b_B] \geq \frac{1}{2} + \frac{\epsilon(n)}{2}.$$

And the protocol is also said to be $\delta(n)$-secure if, for all n and for any PPT adversary Eve, given the security parameter n and the entire transcript (denoted by Trans) between Alice and Bob,

$$\Pr[b' \leftarrow Eve(\mathsf{Trans}, 1^n) : b' = b_A] \leq 1 - \frac{\delta(n)}{2}.$$

If Alice and Bob exchange k messages during the execution of the bit agreement protocol, it is called a k-pass bit agreement. Note that Alice and Bob output the same bit with overwhelming probability, and Eve can compute the bit b_A with only negligible advantage over merely guessing it, as ϵ and δ are overwhelming.

A key agreement protocol is one where two parties, Alice and Bob, given n as a common input, secretly agree on a key in $\{0,1\}^n$ at the end of execution. The key agreement is known to be achieved by polynomially many parallel or sequential executions of the bit agreement protocol if the protocol has a noticeable ϵ and an overwhelming δ [Hol05]. Notice that the parallel repetitions of bit agreements achieve the n-bit key agreement without increasing the round complexity. See [Hol05] and [Hol06] for further details. By one round, we mean a unit process wherein Alice receives, computes and sends a message to Bob, and then Bob receives, computes and sends a message to Alice. Thus, a γ-round key agreement(γ-KA) implies a 2γ-pass key agreement. A γ-round uniform-transcript key agreement(γ-UTKA) is a γ-KA whose transcripts are indistinguishable from uniform distribution.

Definition 12 (γ-round (Uniform-Transcript) Key Agreement Φ (γ-(UT)KA)). *For $\gamma \geq 1$, a γ-round key agreement (exchange) protocol Φ consists of two sub-protocols, Alice (A) and Bob (B), denoted as $\Phi = (\mathsf{A}, \mathsf{B})$. Let α_i and β_i be the ith round messages of A and B respectively. Let $\mathsf{Tran}_i^{\mathsf{A}}$ be the transcript of all the messages up to the ith round from B as $\mathsf{Tran}_i^{\mathsf{A}} = (\beta_1, \beta_2, \ldots, \beta_i)$ and $\mathsf{Tran}_i^{\mathsf{B}} = (\alpha_1, \alpha_2, \ldots, \alpha_i)$. Then, A consists of a family of message-generating algorithms $\mathsf{A}_1, \mathsf{A}_2, \cdots, \mathsf{A}_{\gamma+1}$ defined as follows. For A's random value $r_{\mathsf{A}} \in \{0,1\}^n$:*

$$\mathsf{A}_1 : \{0,1\}^n \to \{0,1\}^n \text{ is defined as } \mathsf{A}_1(r_{\mathsf{A}}) \to \alpha_1,$$
$$\mathsf{A}_i : \{0,1\}^n \times (\{0,1\}^n)^{i-1} \to \{0,1\}^n \text{ is defined as } \mathsf{A}_i(r_{\mathsf{A}}, \mathsf{Tran}_{i-1}^{\mathsf{A}}) \to \alpha_i \text{ for } 2 \leq i \leq \gamma$$
$$\mathsf{A}_{\gamma+1} : \{0,1\}^n \times (\{0,1\}^n)^{\gamma} \to \{0,1\}^n \text{ is defined as } \mathsf{A}_{\gamma+1}(r_{\mathsf{A}}, \mathsf{Tran}_{\gamma}^{\mathsf{A}}) \to s_{\mathsf{A}}.$$

The definition of B is identical to the definition of A except that all A's and α's are replaced with B's and β's. Finally, $\Phi = (\mathsf{A}, \mathsf{B})$ satisfies the following conditions:

1. *For any $\mathcal{A} \in PPT$, given Trans, $\mathcal{A}$ cannot efficiently distinguish the exchanged n-bit key s_{A} from random. Formally, for all random coin tosses r and r' of $\mathcal{A}$ and random inputs r_{A} and r_{B} of A and B,*

$$\Pr[\mathsf{Trans} \leftarrow (\mathsf{A}_{r_{\mathsf{A}}}, \mathsf{B}_{r_{\mathsf{B}}}); s_{\mathsf{A}} \leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{A}}, \mathsf{Trans}_{\gamma}^{\mathsf{A}}) : \mathcal{A}_r(\mathsf{Trans}, s_{\mathsf{A}}) = 1]$$

$$- \Pr[\mathsf{Trans} \leftarrow (\mathsf{A}_{r_{\mathsf{A}}}, \mathsf{B}_{r_{\mathsf{B}}}); \alpha \overset{rand}{\leftarrow} \{0,1\}^n : \mathcal{A}_{r'}(\mathsf{Trans}, \alpha) = 1] \leq \mu(n)$$

   for some negligible function $\mu(n)$.

2. *At the end of execution of* Φ, A *and* B *agree on a secret* s. *Formally, let* $\tau(n)$ *be a negligible function in* n, *then,*

$$\Pr[s_{\mathsf{A}} \leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{A}}, \mathsf{Trans}_{\gamma}^{\mathsf{A}}); s_{\mathsf{B}} \leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{B}}, \mathsf{Trans}_{\gamma}^{\mathsf{B}}) : s_{\mathsf{A}} = s_{\mathsf{B}}] \geq 1 - \tau(n).$$

3. *For the* γ*-round key agreement* Φ *to be a* γ*-round uniform-transcript key agreement, denoted as* Φ_u, *the additional condition below is satisfied. For any PPT adversary* $\mathcal{A}$,

$$\left|\Pr[\mathcal{A}_r(\mathsf{Trans}) = 1] - \Pr[\mathsf{R}_{\gamma} \stackrel{rand}{\leftarrow} (\{0,1\}^n)^{\gamma} : \mathcal{A}_r(\mathsf{R}_{\gamma}) = 1]\right| \leq \epsilon(n),$$

where $\epsilon(n)$ *is a negligible function in* n. B *satisfies the same requirement. That is, no PPT adversary* $\mathcal{A}$ *distinguishes the messages of* A *and* B *from uniform distribution.*

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.

A 2-pass key agreement can be achieved by n parallel repetitions of an underlying 2-pass bit agreement without increasing its round complexity, which we describe as follows. Suppose that we are given a family of DTPs, $(\mathsf{Gen}(\cdot), f, f^{-1})$. Without loss of generality, Alice first chooses a pair of one-way permutation f_k and its inverse permutation $f_{t_k}^{-1}$ by computing a public encryption information k and its private corresponding trapdoor information t_k using $\mathsf{Gen}(\cdot)$. Note that k is computationally indistinguishable from a random string of the same length by the property of DTPs. Alice sends the public key k to Bob. Upon k from Alice, Bob chooses two strings x and r. Bob encrypts x with f_k, so let $y = f_k(x)$. Bob sends y and r to Alice and computes the secret bit $b = <\text{x, r}>$. With y and r, Alice obtains x by inverting y as $x = f_{t_k}^{-1}(y)$. Then, Alice achieves the bit agreement by computing $b = <\text{x, r}>$. Notice that all the messages exchanged between Alice and Bob are either a uniformly random string of length n (i.e., y and r) or pseudo-random strings indistinguishable from uniform (i.e., k). Thus, the above bit agreement is a *uniform-transcript* bit agreement. Hence, the n parallel repetitions of the 2-pass bit agreement achieve a 2-pass n-bit key agreement described in Protocol 1.[1]

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.

[1]We remark that the same randomness r can be used for all of n parallel repetitions of bit agreement instead of using different $r's$ for each of bit agreements. However, this will complicate our exposition later on, so we will use different r's. Clearly this does not affect the security of resulting key agreement protocol.

Alice	Transcript	Bob
$(k, t_k) \leftarrow \mathsf{Gen}(n)$		
	$\xrightarrow{k}$	$x_1, x_2, \cdots, x_n \leftarrow_{\$} \{0,1\}^n$
		$r_1, r_2, \cdots, r_n \leftarrow_{\$} \{0,1\}^n$
		$y_i \leftarrow f_k(x_i)$ for all $i \leq n$
$x_i \leftarrow f_{t_k}^{-1}(y_i)$ for all $i \leq n$	$\xleftarrow{y_1,\cdots,y_n,r_1,\cdots,r_n}$	
$b_i \leftarrow < x_i, r_i >$ for all $i \leq n$		$b_i \leftarrow < x_i, r_i >$ for all $i \leq n$
shared key $sk \leftarrow b_1, b_2, \cdots, b_n$		shared key $sk \leftarrow b_1, b_2, \cdots, b_n$

Protocol 1: 2-pass key agreement based on a DTP (Folklore)

3.1.1 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 (described in Protocol 1) 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.

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

3.1.2 Formal Construction of Non-Adaptively Secure Function F

We first provide the specifications of the underlying primitives used for the construction of F. We have $\tilde{\pi} : K \times (\{0,1\}^n)^{2n+3} \rightarrow (\{0,1\}^n)^{2n+3}$ where K is the key space of $\tilde{\pi}$. $\tilde{\pi}_k$ denotes a PRP with respect to private key k and $\tilde{\pi}_k^{-1}$ is the inversion permutation to $\tilde{\pi}_k$. We are also given PRP $\pi : K \times \{0,1\}^n \rightarrow \{0,1\}^n$. Notice that both $\tilde{\pi}$ and π have the same key space K. Without

loss of generality, K is $\{0,1\}^n$ throughout this paper. Finally, we are given a family of DTPs, $(\mathsf{Gen}(\cdot,\cdot), f, f^{-1})$. We denote f_k and $f_{t_k}^{-1}$ as a permutation and its inverse permutation defined over $\{0,1\}^n$ where (k, t_k) is generated by $\mathsf{Gen}(1^n, r)$ for randomness $r \in \{0,1\}^n$ and $|k| = n$ and $|t_k| = poly(n)$.

Built on the above underlying primitives, the counter-example function F is defined to be from $(\{0,1\}^n)^{2n+3}$ to $(\{0,1\}^n)^{2n+3}$ and internally hides a secret k_{F} in K, which is a private key applied to the underlying primitives in order to generate pseudo-random. Let $I = (u_1, u_2, \ldots, u_{2n+3})$ be an input vector to F where $u_j \in \{0,1\}^n$ for $i = 1, 2, \ldots, 2n+3$. Similarly, $(v_1, v_2, \ldots, v_{2n+3})$ denote an output vector. The formal construction of F is given in Algorithm 1.

Construction of F

1. If $I = (u_1 \neq 0^n, u_2 = 0^n, \cdots, u_{2n+3} = 0^n)$, then
   Output $(v_1, v_2, \ldots, v_{2n+3})$ where
   $(a_1, a_2, \ldots, a_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(0^n, 0^n, \ldots, 0^n)$
   $(x_1, x_2, \ldots, x_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_1, u_2, \ldots, u_{2n+3})$
   Let $y_i = f_{u_1 \oplus a_1}(x_i)$ for $\forall i = 1, 2, \ldots, n$
   $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow (y_1, y_2, \cdots, y_n, x_{n+1}, \ldots, x_{2n+3})$

2. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \ldots, u_{2n+1} \neq 0^n, u_{2n+2} = 0^n, u_{2n+3} = 0^n)$, then
   Output $(v_1, v_2, \ldots, v_{2n+3})$ where
   $(x_1, x_2, \ldots, x_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_{2n+1}, 0^n, \ldots, 0^n)$
   Let $k_i =< x_i, x_{n+i} >$ for $\forall i = 1, 2, \ldots, n$
   Let $k' = k_1 k_2 \ldots k_n$
   $(r_1, r_2, \ldots, r_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_1, u_2, \ldots, u_{2n+3})$
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow (\pi_{k'}(r_1), r_1, \ldots, r_{2n+2})$

3. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \cdots, u_{2n+3} \neq 0^n)$, then
   output $(v_1, v_2, \cdots, v_{2n+3})$ where
   $(x_1, x_2, \cdots, x_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_{2n+1}, 0^n, \cdots, 0^n)$
   Let $k_i =< x_i, x_{n+i} >$ for $\forall i = 1, 2, \cdots, n$
   Let $k' = k_1 k_2 \cdots k_n$
   $(r_1, r_2, \cdots, r_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_1, u_2, \cdots, u_{2n+1}, 0^n, 0^n)$
   $\alpha \leftarrow \pi_{k'}(r_1)$

   (a) If $\pi_{k'}^{-1}(\alpha \oplus u_{2n+2}) = r_1 \oplus u_{2n+3}$,
   then $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow (k_F, 0^n, 0^n, \cdots, 0^n)$

   (b) else$(v_1, v_2, \cdots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_1, u_2, \cdots, u_{2n+3})$

4. If I is not of any previous cases, then
   output $(v_1, v_2, \cdots, v_{2n+3})where$
   $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_F}(u_1, u_2, \cdots, u_{2n+3})$

Algorithm 1: The algorithm of function F

Claim 3.1. *The function* F *is secure against any non-adaptive PPT adversary* $\mathcal{A}(q,t)$*, running in time* t *and making at most* q *non-adaptive queries, where* t *and* q *are any polynomials of security*

parameter n.

Proof. For clarity in the following proof, we denote r_{ij} as the ith randomness at the jth query. To prove the non-adaptive security of F, we will show that

$$\mathbf{Adv}^{\mathsf{F}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{f}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t') + \frac{q}{2^n} \tag{1}$$

where $t' = t + poly(n,q)$, accounting for the extra time costs resulting from our reduction.

Assume that non-adaptive adversary $\mathcal{A}$ chooses q queries as follows. The first query is $Q_1 = (u^*, 0^n, \ldots, 0^n) \in (\{0,1\}^n)^{2n+3}$ and the rest of $q-1$ queries are $Q_i = (u_{i1}, u_{i2}, \ldots, u_{i(2n)}, u^*, 0^n, 0^n)$ for $2 \leq i \leq q$, in which $u_{i1}, u_{i2}, \ldots, u_{i(2n)}$ are arbitrarily chosen for all i. Notice that in cases 1 and 2 of Algorithm 1, once we fix the first coordinate of Q_1 and the $(2n+1)$th coordinate of Q_i's to be equally u^*, we actually fix the shared key k' through all the q queries, so that the first coordinate is an encryption of the second coordinate by $\pi_{k'}$ in the last $q-1$ outputs. Hence, inverting the first n coordinates of output on Q_1 will reveal the key k', and consequently $\mathcal{A}$ distinguishes F from a uniform function R. Since any PPT adversary can invert f_k only with at most negligible probability ϵ_{f_k}, the probability of retrieving k' is at most $(\epsilon_{f_k})^n$, constituting the first term on the right-hand side (RHS) of inequality (1).

Assume that $\mathcal{A}$ makes q queries in the form of $(u_1 \neq 0^n, u_2 \neq 0^n, \cdots, u_{2n+3} \neq 0^n)$, corresponding to case 3 and fixes the first $2n+1$ coordinates of all the queries. So, $\mathcal{A}$ fixes k', r_1, and α in the condition $\pi^{-1}_{k'}(\alpha \oplus u_{2n+2}) = r_1 \oplus u_{2n+3}$. Now, $\mathcal{A}$ only needs to find a pair of u_{2n+2} and u_{2n+3} satisfying the condition so that F reveals its secret key k_{F}. Since π is a permutation, there exists unique u_{2n+3} to each u_{2n+2}, which satisfies the condition. Hence,

$$\Pr[\pi^{-1}_{k'}(\alpha \oplus u_{2n+2}) = r_1 \oplus u_{2n+3} : u_{2n+2}, u_{2n+3} \leftarrow_{\$} \{0,1\}^n] \leq \frac{1}{2^n}.$$

With q queries, $\mathcal{A}$ successfully guesses k_{F} with probability at most $q/2^n$; that is the second term on the RHS of the inequality.

Consider that $\mathcal{A}$ makes q queries such that each input query falls into either case 3 or 4 of Algorithm 1. Since we already showed above that F in the case 3 outputs only pseudo-random strings computed by $\tilde{\pi}_{k_{\mathsf{F}}}$ with an overwhelming probability, we ignore the case that F outputs the secret key k_{F} on one of the q queries. Then, F simply outputs a vector of pseudo-random strings generated by PRP $\tilde{\pi}_{k_{\mathsf{F}}}$ on each input query, which is indistinguishable from uniform randoms of $(\{0,1\}^n)^{2n+3}$. This constitutes $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t')$ in inequality (1).

Consider the case in which $\mathcal{A}$ makes q non-adaptive queries in which one of the queries is $(0^n, 0^n, \cdots, 0^n)$, so it evokes case 4 and the rest of queries evoke case 1 of Algorithm 1. Towards a contradiction, assume that $\mathcal{A}$ distinguishes the outputs corresponding to the q non-adaptive input queries described above. The output of case 4 is indistinguishable from uniform random by the security of PRP $\tilde{\pi}_{k_{\mathsf{F}}}$. This implies that $\mathcal{A}$ distinguishes the outputs of case 1 from uniform randoms. Notice that the first n coordinates of an output of case 1 are strings generated in the following way. $\tilde{\pi}_{k_{\mathsf{F}}}$ on an input query first generates pseudo-random strings and then a trapdoor permutation $f_{u_1 \oplus a_1}$ re-encrypts these pseudo-random strings where $u_1 \oplus a_1$ is known since $\mathcal{A}$ can obtain a_1 from the output of F on $(0^n, 0^n, \cdots, 0^n)$. The rest of coordinates (all coordinates except for the first n coordinates) are strings generated only by $\tilde{\pi}_{k_{\mathsf{F}}}$ on an input query. We recall the description of the output of case 1 on input query $(u_1, u_2, \cdots, u_{2n+3})$ as follows:

$$(\underbrace{f_{u_1 \oplus a_1}(r_1), \cdots, f_{u_1 \oplus a_1}(r_n)}_{\text{first n coordinates}}, \underbrace{r_{n+1}, \cdots, r_{2n+3}}_{\text{the rest of coordinates}})$$

where $(r_1, r_2, \cdots, r_{2n+3}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \cdots, u_{2n+3})$. Since $\mathcal{A}$ distinguishes these outputs of case 1 from uniform randoms, either of the following cases must be true. First, $\mathcal{A}$ distinguishes outputs of case 1 from uniform randoms by distinguishing the first n coordinates of the outputs from uniform randoms. Then, $\mathcal{A}$ can also distinguish outputs of case 4 from uniform randoms as follows. Upon outputs of case 4, $\mathcal{A}$ applies $f_{u_1 \oplus a_1}$ to the first n coordinates of each output and ignores the rest of coordinates of each output. This forces the distribution of the outputs of case 4 to be identical to the distributions of the outputs of case 1 that $\mathcal{A}$ distinguishes from uniform randoms. Therefore, $\mathcal{A}$ distinguishes outputs of case 4 from uniform randoms. This leads to a contradiction to the non-adaptive security of F in case 4 already proven above. Then, it must be true that $\mathcal{A}$ distinguishes outputs of case 1 from uniform randoms by distinguishing the rest of coordinates of outputs from uniform randoms. However, this also enables $\mathcal{A}$ to distinguish outputs of case 4 from uniform randoms by ignoring the first n coordinates of each output. Hence, another contradiction arises to the non-adaptive security of F in case 4. Since we encounter contradictions in both cases, the advantage of $\mathcal{A}$ making q non-adaptive queries of case 1 is also upper-bounded by $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q, t')$ in inequality (1).

Finally, consider that $\mathcal{A}$ makes q queries of case 2 in Algorithm 1. Towards a contradiction, assume that $\mathcal{A}$ distinguishes theses outputs from uniform randoms. Notice that the distribution of the last $2n+1$ coordinates in output vectors (all the elements except for the first two elements) is equivalent to the distribution of the last $2n+1$ coordinates of an output of case 4. This is due to that both distributions are generated by $\tilde{\pi}_{k_{\mathsf{F}}}$ on input queries. We already showed above that if $\mathcal{A}$ distinguishes outputs of case 2 from uniform randoms by distinguishing the last $2n+1$ coordinates from uniform randoms, $\mathcal{A}$ can also distinguish outputs of case 4 from uniform randoms, which leads to a contradiction. This implies that $\mathcal{A}$ distinguishes outputs of case 2 from uniform random by distinguishing the first 2 coordinates from uniform randoms. Hence, distinguishing the outputs of case 2 from uniform randoms over $(\{0,1\}^n)^{2n+3}$ is equivalent to distinguishing

$$(\pi_k(r_1), r_1), (\pi_k(r_2), r_2), \cdots, (\pi_k(r_q), r_q) \tag{2}$$

where $k, r_1, r_2, \cdots r_q$ are uniformly random, from

$$(a_1, b_1), (a_2, b_2), \cdots, (a_q, b_q) \tag{3}$$

where a_i, b_i for all $i = 1, \cdots, q$ are uniformly random.

Let $\mathcal{A}$ distinguish (2) from (3) with a non-negligible probability ξ. We define the ith hybrid distribution H_i as

$$H_i = (\pi_k(r_1), r_1), \cdots, (\pi_k(r_i), r_i), (a_{i+1}, b_{i+1}), \cdots, (a_q, b_q)$$

where k, r_i, a_i, b_i for all $i = 1, \cdots, q$ are uniformly random. Since $|H_0 - H_q| \geq \xi$, there exists i such that $|H_i - H_{i+1}| \geq \xi/q$. Then, we can construct a distinguisher $\mathcal{D}'$ by using $\mathcal{A}$ such that $\mathcal{D}'$ distinguishes π from a random function $\mathsf{R} : \{0,1\}^n \to \{0,1\}^n$ with non-negligible probability as follows. Upon an unknown distribution $(\alpha, \beta) \in (\{0,1\}^n)^2$, $\mathcal{D}'$ generates the ith hybrid distribution as

$$(\pi_k(t_1), t_1), \cdots, (\pi_k(t_{i-1}), t_{i-1}), (\alpha, \beta), (a_{i+1}, b_{i+1}), \cdots, (a_q, b_q)$$

where $t_1, \cdots, t_q, a_{i+1}, b_{i+1}, \cdots, a_q, b_q$ are uniformly random. Then, $\mathcal{D}'$ queries the ith hybrid distribution to $\mathcal{A}$. Since $\mathcal{A}$ distinguishes the ith hybrid distribution with non-negligible probability ξ/q, $\mathcal{D}'$ distinguishes π from R with non-negligible probability ξ/q, which contradicts the indistinguishability of π. This contributes to $\mathbf{Adv}^{\pi}_{\mathcal{A}}(q, t')$ in inequality (1). □

3.1.3 Formal Construction of Non-Adaptively Secure Function G

The function G is also defined from $(\{0,1\}^n)^{2n+3}$ to $(\{0,1\}^n)^{2n+3}$ with a secret k_{G} in K. The notations and standard specifications of underlying primitives remain identical to those for the construction of F in the previous section. The formal construction of G is presented in Algorithm 2.

Claim 3.2. *The function G is secure against any non-adaptive PPT adversary $\mathcal{A}(q,t)$, running in time t and making at most q non-adaptive queries, where t and q are any polynomials of security parameter n.*

Proof. To prove the non-adaptive security of G, we will also show that

$$\mathbf{Adv}^{\mathsf{G}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{f}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t') + \frac{q}{2^n} + \mathbf{Adv}^{\mathsf{Gen}}_{\mathcal{A}}(q,t') \tag{4}$$

where $t' = t + poly(n,q)$ as defined in Claim 3.1 and Gen is a key generation algorithm for a family of DTPs. Since all the cases of G are identical except that the output of G in case (1) is a public key k for trapdoor permutation f, the first four terms on the RHS of (4) are identical to those in Lemma 1. Since the key k is indistinguishable from uniform random over $\{0,1\}^n$ by the property of Gen, any PPT adversary $\mathcal{A}$ distinguishes the key k from uniform random over $\{1,0\}^n$ with only negligible advantage. Hence, the indistinguishability of dense keys constitutes $\mathbf{Adv}^{\mathsf{Gen}}_{\mathcal{A}}(q,t')$ in inequality (4). □

3.1.4 Adaptive Insecurity of Parallel Composition of F and G

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

Proof. To show the claim, we present a particular sequence of four adaptive queries in which the parallel composition $\mathsf{F} \oplus \mathsf{G}$ reveals all the secret keys of F and G, such as k_F and k_G. Let r_{ij} (or s_{ij}) denote the ith randomness of F (or G) upon the jth adaptive query Q_j. Then, our first query Q_1 to $\mathsf{F} \oplus \mathsf{G}(\cdot)$ is $Q_1 = (0^n, 0^n, \cdots, 0^n) \in (\{0,1\})^{2n+3}$. Since the input always (with probability 1) evokes case 4 of Algorithm 1, we have

$$\mathsf{F}(0^n, 0^n, \cdots, 0^n) = \tilde{\pi}_{k_F}(0^n, 0^n, \cdots, 0^n) = (r_{11}, r_{21}, \cdots, r_{(2n+3)1}).$$

In G, since the input falls into case 1 of Algorithm 2, G computes $(s_{11}, s_{21}, \cdots, s_{(2n+3)1})$ by $\tilde{\pi}_{k_G}(0^n, 0^n, \cdots, 0^n)$ and then obtains (k, t_k) by executing $\mathsf{Gen}(1^n, s_{11})$. Finally, G outputs $(k, s_{21}, \cdots, s_{(2n+3)1})$. Thus, the output of $\mathsf{F} \oplus \mathsf{G}(0^n, 0^n, \cdots, 0^n)$ is

$$(r_{11}, r_{21}, \cdots, r_{(2n+3)1}) \oplus (k, s_{21}, \cdots, s_{(2n+3)1}) = (r_{11} \oplus k, r_{21} \oplus s_{21}, \cdots, r_{(2n+3)1} \oplus s_{(2n+3)1}).$$

Obtaining the above output, we define our second adaptive query Q_2 to be:

$$Q_2 = (r_{11} \oplus k, 0^n, 0^n, \cdots, 0^n) \in (\{0,1\}^n)^{2n+3}.$$

Note that Q_2 fails to evoke case 1 of Algorithm 1 when $r_{11} \oplus k = 0^n$. That is, if $r_{11} = k$, then the second adaptive query cannot succeed to succeed to lead F and G into the proper case. The

Construction of G

1. If $I = (u_1 = 0^n, u_2 = 0^n, \cdots, u_{2n+3} = 0^n)$, then
   output $(v_1, v_2, \cdots, v_{2n+3})$ where
   $(s_1, s_2, \cdots, s_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(0^n, \cdots, 0^n)$
   $(k, t_k) \leftarrow \mathsf{Gen}(1^n, s_1)$
   $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow (k, s_2, \cdots s_{2n+3})$

2. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \cdots, u_{2n+1} \neq 0^n, u_{2n+2} = 0^n, u_{2n+3} = 0^n)$, then
   output $(v_1, v_2, \cdots, v_{2n+3})$ where
   $(a_1, a_2, \cdots, a_{2n+3}) \leftarrow \mathsf{G}(0^n, 0^n, \cdots, 0^n)$
   $(k, t_k) \leftarrow \mathsf{Gen}(1^n, a_1)$
   $(b_1, b_2, \cdots, b_{2n+3}) \leftarrow \mathsf{G}(u_{2n+1}, 0^n, \cdots, 0^n)$
   Let $x_i = f_{t_k}^{-1}(u_i \oplus b_i)$ for $i = 1, 2, \cdots, n$
   Let $k_j =< x_i, (u_{n+j} \oplus b_{n+j}) >$ for $j = 1, 2, \cdots, n$
   Let $k' = k_1 k_2 \cdots k_n$
   $(s_1, s_2, \cdots, s_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+3})$
   $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow (\pi_{k'}(s_1), s_1, \cdots, s_{2n+2})$

3. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \cdots, u_{2n+3} \neq 0^n)$, then
   output $(v_1, v_2, \cdots, v_{2n+3})$ where
   $(a_1, a_2, \cdots, a_{2n+3}) \leftarrow \mathsf{G}(0^n, 0^n, \cdots, 0^n)$
   $(k, t_k) \leftarrow \mathsf{Gen}(1^n, a_1)$
   $(b_1, b_2, \cdots, b_{2n+3}) \leftarrow G(u_{2n+1}, 0^n, \cdots, 0^n)$
   Let $x_i = f_{t_k}^{-1}(u_i \oplus b_i)$ for $i = 1, 2, \cdots, n$
   Let $k_j =< x_j, (u_{n+j} \oplus b_{n+j}) >$ for $j = 1, 2, \cdots, n$
   Let $k' = k_1 k_2 \cdots k_n$
   $(c_1, c_2, \cdots, c_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+1}, 0^n, 0^n)$
   $\beta \leftarrow \pi_{k'}(c_1)$

   (a) If $\pi_{k'}^{-1}(\beta \oplus u_{2n+2}) = c_2 \oplus u_{2n+3}$,
   then $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow (0^n, k_G, 0^n, \cdots, 0^n)$

   (b) else $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+3})$

4. If I is not of any previous cases, then
   output $(v_1, v_2, \cdots, v_{2n+3})$ where
   $(v_1, v_2, \cdots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+3})$

Algorithm 2: The algorithm of function G

probability that $r_{11} = k$ is $1/2^n$ which is negligible in n. Therefore, Q_2 successfully evokes case 1 of Algorithm 1 with probability $1 - 1/2^n$.

Upon Q_2 which evokes case 1 of Algorithm 1, F internally simulates the computations of itself on Q_1 to obtain r_{11} by executing $\tilde{\pi}_{k_F}(0^n, 0^n, \cdots, 0^n) = \tilde{\pi}_{k_F}(Q_1) = (r_{11}, r_{22}, \cdots, r_{(2n+3)1})$. Now, F can retrieve the public key k from Q_2 by computing $r_{11} \oplus k \oplus r_{11} = k$. Obtaining k, F computes $y_{i2} = f_k(x_{i2})$ for $i = 1, 2, \cdots, n$ where F computes fresh pseudo-random strings as $\tilde{\pi}_{k_\mathsf{F}}(Q_2) = (x_{12}, x_{22}, \cdots, x_{(2n+3)2})$. Finally, F outputs $(y_{12}, \cdots, y_{n2}, x_{(n+1)2}, \cdots, x_{(2n+3)2})$.

In G, the input Q_2 is of case 4 of Algorithm 2. Hence, G outputs fresh pseudo-randoms as follows.

$$\mathsf{G}(r_{11} \oplus k, 0^n, \cdots, 0^n) = \tilde{\pi}_{k_G}(r_{11} \oplus k, 0^n, \cdots, 0^n) = (s_{12}, s_{22}, \cdots, s_{(2n+3)2}) \tag{5}$$

Thus, we have

$$\begin{aligned}(\mathsf{F} \oplus \mathsf{G})(Q_2) &= (y_{12}, \cdots, y_{n2}, x_{(n+1)2}, \cdots, x_{(2n+3)2}) \oplus (s_{12}, s_{22}, \cdots, s_{(2n+3)2}) \\ &= (y_{12} \oplus s_{12}, \cdots, y_{n2} \oplus s_{n2}, x_{(n+1)2} \oplus s_{(n+1)2}, \cdots, x_{(2n+3)2} \oplus s_{(2n+3)2}).\end{aligned}$$

We define our third adaptive query $Q_3 \in (\{0,1\}^n)^{2n+3}$ to be

$$Q_3 = (y_{12} \oplus s_{12}, \cdots, x_{(2n)2} \oplus s_{(2n)2}, r_{11} \oplus k, 0^n, 0^n).$$

Assuming that Q_2 succeeds to evoke case 1 of Algorithm 1 and case 4 of Algorithm 2 (Recall that Q_1 always succeeds.), the third adaptive query Q_3 succeeds to evoke case 2 of Algorithm 1 and Algorithm 2 only when none of the first $2n+1$ coordinates of Q_3 is 0^n. Since the $(2n+1)$th coordinate (i.e., $r_{11} \oplus k$) is taken from Q_2, the $(2n+1)$th coordinate is guaranteed not to be 0^n. Hence, for Q_3 to be a valid adaptive query, it must be the case that $y_{12} \neq s_{12}$, $\cdots$, $x_{(2n)2} \neq s_{(2n)2}$. In other words, Q_3 fails if at least one of equalities $y_{12} = s_{12}$, $\cdots$, $x_{(2n)2} = s_{(2n)2}$ occurs. Each of the equalities occurs with probability $1/2^n$ so that the total failing probability of Q_3 is $2n/2^n$ which is negligible in n. Thus, Q_3 succeeds to evoke case 2 of Algorithm 1 and Algorithm 2 with probability $1 - 2n/2^n$.

Then F upon Q_3 computes,

$$\tilde{\pi}_{k_F}(u_{2n+1}, 0^n, \cdots, 0^n) = \tilde{\pi}_{k_F}(r_{11} \oplus k, 0^n, \cdots, 0^n) = \tilde{\pi}_{k_F}(Q_2) = (x_{12}, x_{22}, \cdots, x_{(2n+3)2}).$$

Then, F computes n hard-core bits by computing

$$k_i =< x_{i2}, x_{(n+i)2} > \text{ for } \forall i = 1, 2, \cdots, n. \tag{6}$$

So, F obtains a shared key $k' = k_1 k_2 \cdots k_n$. Finally, F computes fresh pseudo-random as

$$\tilde{\pi}_{k_F}(u_1, u_2, \cdots, u_{2n+3}) = (r_{13}, r_{23}, \cdots, r_{(2n+3)3}),$$

and outputs $(\pi_{k'}(r_{13}), r_{13}, \cdots, r_{(2n+2)3})$.

As Q_3 evokes case 2 of Algorithm 2, G retrieves (k, t_k) by computing $\mathsf{Gen}(1^n, s_{11})$ where $(s_{11}, s_{21}, \cdots, s_{(2n+3)1}) = \tilde{\pi}_{k_\mathsf{G}}(Q_1)$. Then, G proceeds to compute the followings:

$$\begin{aligned}\mathsf{G}(u_{2n+1}, 0^n, \cdots, 0^n) &= \mathsf{G}(r_{11} \oplus k, 0^n, \cdots, 0^n) \\ &= (a_1, a_2, \cdots, a_{2n+3}) \\ &= (s_{12}, s_{22}, \cdots, s_{(2n+3)2}) \text{ by (5)}.\end{aligned}$$

Using the trapdoor t_k, G computes

$$x_i = f_{t_k}^{-1}(u_i \oplus a_1) = f_{t_k}^{-1}(y_{i2} \oplus s_{i2} \oplus s_{i2}) = f_{t_k}^{-1}(y_{i2}) = x_{i2} \text{ for } \forall i = 1, 2, \cdots, n.$$

Then, G can compute for $\forall j = 1, 2, \cdots, n$

$$\begin{aligned} k_j &=< x_j, (u_{n+j} \oplus a_{n+j}) > \\ &=< x_{j2}, (x_{(n+j)2} \oplus s_{(n+j)2} \oplus s_{(n+j)2}) > \\ &=< x_{j2}, x_{(n+j)2} > . \end{aligned}$$

G constructs a shared key k' by letting $k' = k_1 k_2 \cdots k_n$. Notice that x_{j2} and $x_{(n+j)2}$ are respectively equivalent to x_{i2} and $x_{(n+i)2}$ in F's computation at (6). Thus, G's $k' = \mathsf{F}$'s k'. Finally, G computes fresh pseudo-randoms as

$$\tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+3}) = (s_{13}, s_{23}, \cdots, s_{(2n+3)3}),$$

and outputs $(\pi_{k'}(s_{13}), s_{13}, \cdots, s_{(2n+2)3})$. Therefore, the output of $\mathsf{F} \oplus \mathsf{G}$ on the third input is

$$(\pi_{k'}(r_{13}) \oplus \pi_{k'}(s_{13}), r_{13} \oplus s_{13}, \cdots, r_{(2n+3)3} \oplus s_{(2n+3)3}).$$

Our final adaptive input Q_4 to $\mathsf{F} \oplus \mathsf{G}$ is

$$Q_4 = (y_{12} \oplus s_{12}, \cdots, y_{n2} \oplus s_{n2}, x_{(n+1)2} \oplus s_{(n+1)2}, \cdots, x_{(2n)2} \oplus s_{(2n)2}, r_{11} \oplus s_{11}, \pi_{k'}(r_{13}) \oplus \pi_{k'}(s_{13}), r_{13} \oplus s_{13}).$$

Conditioned that all the previous adaptive queries are successful, consider the probability that Q_4 succeeds to evoke case 3 of Algorithm 1 and Algorithm 2. That is, it is the probability that none of the coordinates of Q_4 is 0^n. Notice that the first $2n + 1$ coordinates are taken from Q_3 (i.e., the first $2n$ coordinates) and Q_2 (i.e., the $(2n + 1)$th coordinate). Hence, none of the first $2n + 1$ coordinates are guaranteed to be 0^n as we conditioned that Q_2 and Q_3 are successful adaptive queries. Q_4 fails if $\pi_{k'}(r_{13}) = \pi_{k'}(s_{13})$ or $r_{13} = s_{13}$ in which each of the cases occurs with probability $1/2^n$. Therefore, Q_4 succeeds to evoke case 3 of Algorithm 1 and Algorithm 2 with probability $1 - 2/2^n$.

On Q_4, which evokes case 3 of Algorithm 1, F retrieves the same shared key k' as in (6) since F only requires u_{2n+1} to be $r_{11} \oplus k$ as in the previous round. Obtaining k', F simulate the computations of the third round. In particular, F computes

$$\tilde{\pi}_{k_F}(u_1, u_2, \cdots, u_{2n+1}, 0^n, 0^n) = \tilde{\pi}_{k_F}(Q_3) = (r_{13}, r_{23}, \cdots, r_{(2n+3)3}).$$

Since $\alpha = \pi_{k'}(r_{13})$,

$$\begin{aligned} \pi^{-1}_{k'}(\alpha \oplus u_{2n+2}) &= \pi^{-1}_{k'}(\pi_{k'}(r_{13}) \oplus \pi_{k'}(r_{13}) \oplus \pi_{k'}(s_{13})) \\ &= \pi^{-1}_{k'}(\pi_{k'}(s_{13})) \\ &= s_{13} \\ &= r_{13} \oplus r_{13} \oplus s_{13} \\ &= r_{13} \oplus u_{2n+3}. \end{aligned}$$

Therefore, F outputs $(k_{\mathsf{F}}, 0^n, \cdots, 0^n)$.

Q_4 evokes in case 3 of Algorithm 2. Again, G starts the fourth round computation by retrieving (k, t_k) as it computes $\mathsf{Gen}(1^n, s_{11})$ where $(s_{11}, s_{21}, \cdots, s_{(2n+3)1}) = \tilde{\pi}_{k_{\mathsf{G}}}(Q_1)$. Then, similarly to F, G also computes the shared key k' by using the first $2n$ elements of Q_4, which are equivalent to the first $2n$ coordinates of Q_3. Thus, G retrieves the shared key k'. Then, G proceeds to check if the equality in case 3.(a) holds as follows:

$$\tilde{\pi}_{k_G}(u_1, u_2, \cdots, u_{2n+1}, 0^n, 0^n) = \tilde{\pi}_{k_G}(Q_3) = (b_1, b_2, \cdots, b_{2n+3}) = (s_{13}, s_{23}, \cdots, s_{(2n+3)3}).$$

Since $\beta = \pi_{k'}(s_{13}) = \pi_{k'}(b_1)$,

$$\begin{aligned}\pi_{k'}^{-1}(\beta \oplus u_{2n+2}) &= \pi_{k'}^{-1}(\pi_{k'}(s_{13}) \oplus \pi_{k'}(r_{13}) \oplus \pi_{k'}(s_{13})) \\ &= \pi_{k'}^{-1}(\pi_{k'}(r_{13})) \\ &= r_{13} \\ &= s_{13} \oplus s_{13} \oplus r_{13} \\ &= b_{13} \oplus u_{2n+3}.\end{aligned}$$

Consequently, G outputs $(0^n, k_{\mathsf{G}}, 0^n, \cdots, 0^n)$. Therefore, the output of $\mathsf{F} \oplus \mathsf{G}$ on Q_4 is

$$(k_{\mathsf{F}}, 0^n, \cdots, 0^n) \oplus (0^n, k_{\mathsf{G}}, 0^n, \cdots, 0^n) = (k_{\mathsf{F}}, k_{\mathsf{G}}, 0^n, \cdots, 0^n),$$

which reveals all of the secret keys of F and G. □

By Claim 3.1, 3.2, and 3.3, we immediately obtains the following lemma:

Lemma 1. *Suppose that a dense trapdoor permutation exists. Then, there exist non-adaptively secure functions* F *and* G *whose parallel composition* $\mathsf{F} \oplus \mathsf{G}$ *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 *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 non-adaptively secure.

3.2.1 Intuitions of Sequential Composition of F and G

We provide the intuitive description of our counter-example functions F and G for which we provide the high-level overview of their formal constructions. 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_{F}' (resp. k_{G} and k_{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_{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:

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

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 $1 \leq i \leq n$. 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 $1 \leq i \leq n$. 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'_{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.

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

3.2.2 Formal Construction of Non-Adaptively Secure Function F

For the underlying primitives and their standard notations, recall the PRPs π_k, $\tilde{\pi}_k$, and a family of DTP $(\mathsf{Gen}(\cdot,\cdot), f, f^{-1})$ given in Section 3.1.2 of adaptively insecure parallel composition based on DTP. Function F is defined over $(\{0,1\}^n)^{2n+3}$. F internally contains two n-bit secret seeds k_{F} and k'_{F}. The formal construction of F is given in Algorithm 3.

Construction of F

1. If $I = (u_1 = 0^n, u_2 = 0^n, \ldots, u_{2n+3} = 0^n)$, then
   output $(v_1, v_2, \ldots, v_{2n+3})$ where
   $(r_1, r_2, \ldots, r_{2n+3}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \ldots, u_{2n+3})$
   $(k, t_k) \leftarrow \mathsf{Gen}(1^n, r_1)$
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow (k, r_2, \ldots, r_{2n+3})$
2. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \ldots, u_{2n+3} \neq 0^n)$, then
   $(a_1, a_2, \cdots, a_{2n+3}) \leftarrow \tilde{\pi}_{k_G}(0^n, \cdots, 0^n)$
   $(k, t_k) \leftarrow \mathsf{Gen}(1^n, a_1)$
   Let $k' = k_1 k_2 \ldots k_n$ where $k_i =< f_{t_k}^{-1}(u_i), u_{n+i} >$ for all $i = 1 \ldots n$
   (a) If $u_{2n+1} = \pi_{k'}(u_{2n+2})$, then
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow (\pi_{k'}^{-1}(u_{2n+3}), k_{\mathsf{F}}, k'_{\mathsf{F}}, 0^n, \ldots, 0^n)$
   (b) else
   output $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \ldots, u_{2n+3})$
3. If I is not of any previous cases
   output $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \ldots, u_{2n+3})$

Algorithm 3: The algorithm of function F

Claim 3.4. *The function* F *is secure against any non-adaptive PPT adversary* $\mathcal{A}(q,t)$*, running in time* t *and making at most* q *non-adaptive queries, where* t *and* q *are any polynomials of security parameter* n*.*

Proof. We prove this claim by showing that

$$\mathbf{Adv}^{\mathsf{F}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\mathsf{Gen}}_{\mathcal{A}}(q,t') + \frac{q}{2^n} \tag{7}$$

where $t' = t + poly(n,q)$, which accounts for the extra time costs resulting from our reduction. Let $Q_i \in (\{0,1\}^n)^{2n+3}$ for $i = 1,\ldots,q$ denote the ith query that PPT non-adaptive adversary $\mathcal{A}$ generates. Then, towards a contradiction, assume that $\mathcal{A}$ distinguishes the outputs of F on Q_i's from uniform random.

We first consider where every Q_i falls into case 3 of Algorithm 3. Since the output of these two cases is directly computed from $\tilde{\pi}$, distinguishing the outputs of these cases implies that $\mathcal{A}$ effectively distinguishes $\tilde{\pi}$ from uniform random functions; this leads to a contradiction. Hence, the distinguishing advantage of $\mathcal{A}$ over F is upper-bounded by the distinguishing advantage of $\mathcal{A}$ over $\tilde{\pi}$, which constitutes term $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t')$ in inequality (7).

Consider the case in which every Q_i falls into case 1. Then, every coordinate except the first coordinate in output vectors is generated by $\tilde{\pi}(Q_i)$. If $\mathcal{A}$ distinguishes F from uniform functions through distinguishing those coordinates generated by $\tilde{\pi}(Q_i)$ from uniform random over $(\{0,1\}^n)^{2n+2}$, then $\mathcal{A}$ clearly can distinguish F from uniform random functions with non-adaptive queries of case 3; this raises a contradiction to the non-adaptive security of F on queries of case 3 proven above. Hence, $\mathcal{A}$ distinguishes F from uniform random functions by distinguishing the first coordinates of outputs from uniform random over $\{0,1\}^n$. However, Gen generates polynomially dense public key for a trapdoor one-way permutation f, the first coordinate k of each output vector is indistinguishable from uniform random over $\{0,1\}^n$; this leads to another contradiction. Therefore, this contributes to term $\mathbf{Adv}^{\mathsf{Gen}}_{\mathcal{A}}(q,t')$ in inequality (7).

Finally, suppose that $\mathcal{A}$ generates q non-adaptive queries of the form $Q_i = (u_1 \neq 0^n, u_2 \neq 0^n, \ldots, u_{2n+3} \neq 0^n)$, which is of case 2 of Algorithm 3. Unless one of Q_i's satisfies $u_{2n+1} = \pi_{k'}(u_{2n+2})$, all the outputs of F are simply computed by $\tilde{\pi}$. Hence, the distinguishing advantage of $\mathcal{A}$ with Q_i's is upper-bounded by $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t')$ when none of Q_i's satisfies $u_{2n+1} = \pi_{k'}(u_{2n+2})$. To succeed to satisfy the condition of $u_{2n+1} = \pi_{k'}(u_{2n+2})$, the adversary $\mathcal{A}$ can make an arbitrary pick on u_{2n+2} in one of Q_i's such that $u_{2n+1} = \pi_{k'}(u_{2n+2})$. The probability of finding is $q/2^n$ which appears in inequality (7). □

3.2.3 Formal Construction of Non-Adaptively Secure Function G

Similarly to function F, function G is also defined over $(\{0,1\}^n)^{2n+3}$ and contains two n-bit secret keys k_{G} and k'_{G}. The underlying primitives and standard notations are identical to those used to construct F in the previous section. The formal definition of function G is provided in Algorithm 4.

Claim 3.5. *The function* G *is secure against any non-adaptive PPT adversary* $\mathcal{A}(q,t)$*, running in time* t *and making at most* q *non-adaptive queries, where* t *and* q *are any polynomials of security parameter* n*.*

Proof. To prove the claim, we will show the following inequality

$$\mathbf{Adv}^{\mathsf{G}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{f}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t') + \frac{q}{2^n} \tag{8}$$

where $t' = t + poly(n,q)$, which accounts for the extra time costs resulting from our reduction. Let $Q_i = (u_{i1}, u_{i2}, \cdots, u_{i(2n+3)}) \in (\{0,1\}^n)^{2n+3}$ for $i = 1,2,\ldots,q$ denote the ith query that PPT

Construction of G

1. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, \ldots, u_{2n+3} \neq 0^n)$, then
   output $(v_1, v_2, \ldots, v_{2n+3})$ where
   $(s_1, s_2, \ldots, s_{2n+3}) \leftarrow \tilde{\pi}_{k_\mathsf{G}}(u_1, u_2, \ldots, u_{2n+3})$
   Let $k' = k_1 k_2 \ldots k_n$ where $k_i = < s_i, s_{n+i} >$ for all $i = 1 \ldots n$
   $y_i = f_{u_1}(s_i)$ for all $i = 1 \ldots n$
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow (y_1, \ldots, y_n, s_{n+1}, \ldots, s_{2n}, \pi_{k'}(s_{2n+1}), s_{2n+1}, \pi_{k'}(k'_\mathsf{G}))$
2. If $I = (u_1 \neq 0^n, u_2 \neq 0^n, u_3 \neq 0^n, u_4 = 0^n, \ldots, u_{2n+3} = 0^n)$ then
   output $(v_1, v_2, \ldots, v_{2n+3})$ where
   (a) If $u_1 = k'_\mathsf{G}$, then
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow (k_\mathsf{G}, k'_\mathsf{G}, u_2, u_3, 0^n, \ldots, 0^n)$
   (b) else
   $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_\mathsf{G}}(u_1, u_2, \ldots, u_{2n+3})$
3. If I is not of any previous cases
   output $(v_1, v_2, \ldots, v_{2n+3}) \leftarrow \tilde{\pi}_{k_\mathsf{G}}(u_1, u_2, \ldots, u_{2n+3})$

Algorithm 4: The algorithm of function G

non-adaptive adversary $\mathcal{A}$ generates. Towards a contradiction, assume that PPT non-adaptive adversary $\mathcal{A}$ distinguishes G from a uniform random function.

Because the output of G in case 2(b) and 3 of Algorithm 4 is simply computed from $\tilde{\pi}_{k_\mathsf{G}}(Q_i)$, distinguishing the outputs of those cases from uniform random is identical to distinguishing the outputs of $\tilde{\pi}(Q_i)$ from uniform random. This leads to a contradiction to the adaptive security of $\tilde{\pi}$. Therefore, inequality (8) counts this case with term $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q, t')$.

On the other hand, the $q/2^n$ term in (8) captures the probability that adversary $\mathcal{A}$ correctly guesses key k'_G and put k'_G to be the first coordinate of one of q non-adaptive queries which evoke case 2(a).

Finally, consider the case that $\mathcal{A}$ generates q non-adaptive queries evoking case 1 in Algorithm 4. The first $2n$ coordinates and the $(2n+2)$th coordinate of an output of the case are generated by $\tilde{\pi}_{k_\mathsf{G}}(Q_i)$. In fact, the first n coordinates are generated as follows. $\tilde{\pi}_{k_\mathsf{G}}(Q_i)$ first generates $2n+3$ pseudo-random strings and then a trapdoor permutation $f_{u_{i1}}$ encrypts the first n coordinates for all i. We recall the description of outputs of G on Q_i's evoking case 1 of Algorithm 4 below:

$$\mathsf{G}(Q_i) = (\underbrace{f_{u_{i1}}(s_{i1}), \ldots, f_{u_{i1}}(s_{in})}_{\text{the first n coordinates}}, \underbrace{s_{i(n+1)}, \ldots, s_{i(2n)}}_{\text{the second n coordinates}}, \underbrace{\pi_{k'}(s_{i(2n+1)}), s_{i(2n+1)}, \pi_{k'}(k'_\mathsf{G})}_{\text{the last three coordinates}}) \quad (9)$$

where $(s_{i1}, s_{i2}, \cdots, s_{i(2n+3)}) \leftarrow \tilde{\pi}_{k_\mathsf{G}}(Q_i)$.

Notice that inverting the first n pseudo-random strings encrypted by $f_{u_{i1}}(\cdot)$ reveals the private key k' which enables $\mathcal{A}$ to obtain $k_{\mathsf{G}'}$ by inverting the last coordinate $\pi_{k'}(k_\mathsf{G})$ of outputs. Since the advantage of inverting a trapdoor one-way permutation without trapdoor information $t_{u_{i1}}$ is negligible, this contributes to $\mathbf{Adv}^{f}_{\mathcal{A}}(q, t')$ in inequality (8).

In addition, suppose that $\mathcal{A}$ distinguishes G from uniform random functions by distinguishing the first $2n$ coordinates and the $(2n+2)$th coordinate from uniform random over $(\{0,1\}^n)^{2n+1}$. Then, $\mathcal{A}$ effectively distinguishes G from uniform random functions even with q non-adaptive queries evoking case 3 by applying $f_{u_{i1}}(\cdot)$ to the first n coordinates of each output of case 3 and ignoring

the $(2n+1)$th and $(2n+3)$th coordinates of each output. This is clearly a contradiction to the non-adaptive security of G in case 3, which is proven above. Therefore, this accounts for term $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t')$ in inequality (8).

Now, consider the case that $\mathcal{A}$ distinguishes G of case 1 from uniform random functions by distinguishing the last three coordinates in (9) from uniform random over $(\{0,1\}^n)^3$. The final coordinate is an encryption of k'_{G}, one of the secret keys of G by $\pi_{k'}(\cdot)$ where k' is hidden by a trapdoor permutation. Notice in Algorithm 4 that k_{G} is the only secret key used to generate pseudo-random strings and is independent of k'_{G}. Hence, by letting G have two secret keys, G avoids issues relevant to *key-dependent message security* and *circular-security* [CL01, BRS02, BHHO08]. Thus, distinguishing G from uniform random functions via the last three coordinates of outputs is equivalent to distinguishing

$$(\pi_{k_1}(s_1), s_1, \pi_{k_1}(k)), (\pi_{k_2}(s_2), s_2, \pi_{k_2}(k)), \ldots, (\pi_{k_q}(s_q), s_q, \pi_{k_q}(k)) \tag{10}$$

from

$$(a_1, b_1, c_1), (a_2, b_2, c_2), \ldots, (a_q, b_q, c_q) \tag{11}$$

where $a_1, b_1, c_1, \cdots, a_q, b_q, c_q$, $s_1, s_2, \cdots, s_q$ and $k_1, k_2, \cdots, k_q, k$ are uniformly chosen. We consider $s_1, s_2, \cdots, s_q$ and $k_1, k_2, \cdots, k_q$ as uniform random rather than pseudo-random since we already proved that $s_1, s_2, \cdots, s_q$ are indistinguishable from uniform random by $\mathcal{A}$ and finding $k_1, k_2, \cdots, k_q$ by $\mathcal{A}$ succeeds with only negligible probability. Let $\mathcal{A}$ distinguish (10) from (11) with non-negligible probability ξ. Using $\mathcal{A}$ as a subroutine, we will construct another PPT distinguisher $\mathcal{D}$ which distinguishes PRP π from a uniform random function. Let the distribution of the ith hybrid be defined by:

$$H_i = (\pi_{k_1}(s_1), s_1, \pi_{k_1}(k)), \cdots, (\pi_{k_i}(s_i), s_i, \pi_{k_i}(k)), (a_{i+1}, b_{i+1}, c_{i+1}), \ldots, (a_q, b_q, c_q).$$

Since $\mathcal{A}$ distinguishes (10) from (11) with non-negligible probability ξ, there exists j such that $|H_j - H_{j+1}| \geq \xi/q$. Given unknown distribution $(x, y, z) \in (\{0,1\}^n)^3$, $\mathcal{D}$ generates a new jth hybrid distribution as

$$(\pi_{k_1}(s_1), s_1, \pi_{k_1}(k)), \cdots, (\pi_{k_{j-1}}(s_{j-1}), s_{j-1}, \pi_{k_{j-1}}(k)), (x, y, z), (a_{j+1}, b_{j+1}, c_{j+1}), \ldots, (a_q, b_q, c_q).$$

Then, $\mathcal{D}$ feed this hybrid distribution to $\mathcal{A}$ which distinguishes H_j from H_{j+1} with non-negligible distinguishing advantage ξ/q. Hence, $\mathcal{D}$ correctly answers whether (x, y, z) is from (10) or (11) with non-negligible distinguishing advantage ξ/q. This implies that $\mathcal{D}$ distinguishes π from uniform random functions with non-negligible distinguishing advantage ξ/q, which is a contradiction. Therefore, this contributes to term $\mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t')$ in inequality (8). □

3.2.4 Adaptive Insecurity of Sequential Composition of F and G

Claim 3.6. *The sequential composition $\mathsf{G}(\mathsf{F}(\cdot))$ is breakable by two adaptive queries.*

Proof. We will present two adaptive queries which reveals all the secrets of F and G (i.e., k_{F} and k_{G}). In the following proof, let r_{ij} denote the ith randomness of F generated in jth query, and s_{ij} denote the ith randomness of G on the jth adaptive query.

We define the first adaptive query Q_1 to be $(0^n, \ldots, 0^n)$ which evokes case 1 of Algorithm 3. $\mathsf{F}(Q_1)$ first computes $(r_{11}, r_{21}, \cdots, r_{(2n+3)1})$ by executing $\tilde{\pi}_{k_{\mathsf{F}}}(Q_1)$ and then computes (k, t_k) where $(k, t_k) \leftarrow \mathsf{Gen}(1^n, r_{11})$. Consequently, $\mathsf{F}(Q_1)$ is $(k, r_{21}, \cdots, r_{(2n+3)1})$.

Upon $\mathsf{F}(Q_1) = (k, r_{21}, \cdots, r_{(2n+3)1})$, G enters case 1 of Algorithm 4 with overwhelming probability. In particular, $(k, r_{21}, \cdots, r_{(2n+3)1})$ must not contain any 0^n for G to enter case 1. Since

each coordinate can be 0^n with probability $1/2^n$, the probability that G fails to enter case 1 is $(2n+3)/2^n$ which is negligible for sufficiently large n. Then, G computes fresh pseudo-randoms $(s_{11}, s_{21}, \cdots, s_{(2n+3)1})$ by executing $\tilde{\pi}_{k_\mathsf{G}}(\mathsf{F}(Q_1))$. G obtains a shared key k' as $k' = k_1 k_2 \cdots k_n$ where $k_i =< s_i, s_{n+i} >$ for $1 \leq i \leq n$. Hence, G outputs

$$(y_1, y_2, \cdots, y_n, s_{(n+1)1}, s_{(n+2)1}, \cdots, s_{(2n)1}, \pi_{k'}(s_{(2n+1)1}), s_{(2n+1)1}, \pi_{k'}(k_\mathsf{G})), \tag{12}$$

where y_i is $\pi_k(s_{i1})$ for $1 \leq i \leq n$.

We defined our second (last) adaptive query Q_2 to be simply the above output of the first round computation as in (12):

$$Q_2 = (y_1, y_2, \cdots, y_n, s_{(n+1)1}, s_{(n+2)1}, \cdots, s_{(2n)1}, \pi_{k'}(s_{(2n+1)1}), s_{(2n+1)1}, \pi_{k'}(k_\mathsf{G}^{'})).$$

Notice that Q_2 evokes case 2 of Algorithm 3 with overwhelming probability. To evoke case 2, Q_2 is required not to contain any 0^n. Since the probability is $1/2^n$ that one coordinate is 0^n, the total probability for Q_2 to fail to evoke case 2 is $(2n+3)/2^n$ which is negligible for sufficiently large n. Entering case 2, F first obtains (k, t_k) as it computes $\mathsf{Gen}(1^n, r_{11})$ where r_{11} can be obtained by $\tilde{\pi}_{k_\mathsf{F}}(Q_1)$. Recall that Q_1 is a fixed zero vector so that F can internally compose Q_1. F obtains s_{i1} by computing $f_{t_k}^{-1}(y_i)$ for $1 \leq i \leq n$. So, F retrieves the shared key $k' = k_1 k_2 \cdots k_n$ where $k_i =< s_i, s_{n+i} >$ for $1 \leq i \leq n$. Obtaining k', F can find out that the condition of case 2.(b) holds by inverting $\pi_{k'}(s_{(2n+1)1})$. Thus, F is convinced that Q_2 is an adaptively generated query so that F computes the inverse of the last coordinate in Q_2 to obtain k_G (i.e., $\pi_{k'}^{-1}(\pi_{k'}(k_\mathsf{G}^{'})) = k_\mathsf{G}^{'}$). Finally, F outputs $(k_\mathsf{G}^{'}, k_\mathsf{F}, k_\mathsf{F}^{'}, 0^n, \cdots, 0^n)$.

Upon $(k_\mathsf{G}^{'}, k_\mathsf{F}, k_\mathsf{F}^{'}, 0^n, \cdots, 0^n)$ which leads G to case 2, G also is convinced that the input query is adaptively generated since the first coordinate is $k_\mathsf{G}^{'}$. Notice that $(k_\mathsf{G}^{'}, k_\mathsf{F}, k_\mathsf{F}^{'}, 0^n, \cdots, 0^n)$ will successfully evokes case 2 unless $k_\mathsf{G}^{'}$, k_F, or $k_\mathsf{F}^{'}$ is 0^n. Hence the probability is negligible $3/2^n$ that G fails to enter case 2. Therefore, G simply outputs $(k_\mathsf{G}, k_\mathsf{G}^{'}, k_\mathsf{F}, k_\mathsf{F}^{'}, 0^n, \cdots, 0^n)$, which reveals all the secrets of F and G. □

Proving Claim 3.4, 3.5, and 3.6 substantiates the following lemma.

Lemma 2. *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.*

Therefore, by Lemma 1 and 2, we immediately obtains the following theorem which concludes the impossibility of adaptively secure composition under the existence of DTP.

Theorem 6. *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 [Pie06]. 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 using the similar technique to that in [Pie06].

4.1 Parallel Composition Insecurity vs. Uniform Transcript Key Agreement

4.1.1 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 [Pie06]. That is, for $k \geq 2$, 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 [Pie06], we construct a $(2k-1)$-pass uniform-transcript bit agreement (UTBA) 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* [Hol05]. 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 UTBA from two adaptively pseudo-random functions whose parallel composition is not k-adaptively secure when $k = 2$ in Protocol 2. The 3-pass UTBA in Protocol 2 may be easily extended to the $(2k-1)$-pass UTBA for arbitrary k

Protocol Bit-Agreement(1^n)

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

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

and general adaptive distinguisher $\mathcal{D}$. We describe the above protocol and the extension in detail in the proof of following theorem.

Theorem 7. *Let* F *and* G *be* $(k-1)$*-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 for* $k \geq 2$*.*

Proof. We present the special case where $k = 2$. Then, we explain how to generalize the technique for arbitrary k. Let F and G be non-adaptively (2-adaptively) secure pseudo-random functions from $K \times \{0,1\}^n \to \{0,1\}^n$ where K is the key space of F and G. Without loss of generality, we let K be $\{0,1\}^n$. Also, let $\mathsf{Gen}_{\mathsf{F}}$ and $\mathsf{Gen}_{\mathsf{G}}$ be the key generation algorithms from $\{1\}^l$ to $\{0,1\}^l$ defined as $\mathsf{Gen}_{\mathsf{F}}(1^l) \to x$ for $x \in \{0,1\}^l$. In this proof, $l = n$ since $K = \{0,1\}^n$.

In Protocol 2, $\mathcal{D}$ is a 2-adaptive distinguisher distinguishing $\mathsf{F} \oplus \mathsf{G}$ from a (uniform) random function $\mathsf{R}^n : \{0,1\}^n \to \{0,1\}^n$. Without loss of generality, $\mathcal{D}$, upon an input (y_1, y_2) for $y_1, y_2 \in \{0,1\}^n$, outputs 1 if and only if $\mathcal{D}$ determines that the input is the output of a uniform random of

length n. Now we want to show that the protocol Bit-Agreement(1^n) in Protocol 2 has correlation $\epsilon(n)$ and is secure with probability $\delta(n)$ where ϵ is non-negligible and δ is overwhelming given the security parameter n. Furthermore, we want to prove that Bit-Agreement(1^n) satisfies the property of uniform-transcript. Therefore, n parallel repetitions of Bit-Agreement(1^n) will achieve the n-bit

Claim 4.1 (Non-negligible Correctness)**.** *The protocol Bit-Agreement*(1^n) *has non-negligible* ϵ*.*

Proof. Let b_1, b_2, and b_3 be bits defined as in the following three cases.

case 1	case 2	case 3
$b_1 \leftarrow \mathcal{D}(y_1, y_2)$ where,	$b_2 \leftarrow \mathcal{D}(y_1, y_2)$ where,	$b_3 \leftarrow \mathcal{D}(y_1, y_2)$ where,
$k_1 \leftarrow \mathsf{Gen}_{\mathsf{F}}(1^n)$	$x_1 \leftarrow \mathcal{D}(1^n)$	$y_1 \leftarrow_{\$} \{0,1\}^n$
$k_2 \leftarrow \mathsf{Gen}_{\mathsf{G}}(1^n)$	$y_1 \leftarrow \mathsf{R}^n(x_1)$	$y_2 \leftarrow_{\$} \{0,1\}^n$
$x_1 \leftarrow \mathcal{D}(1^n)$	$x_2 \leftarrow \mathcal{D}(y_1)$	
$y_1 \leftarrow \mathsf{F}_{k_1}(x_1) \oplus \mathsf{G}_{k_2}(x_1)$	$y_2 \leftarrow \mathsf{R}^n(x_2)$	
$x_2 \leftarrow \mathcal{D}(y_1)$		
$y_2 \leftarrow \mathsf{F}_{k_1}(x_2) \oplus \mathsf{G}_{k_2}(x_2)$		

$\mathcal{D}$ is a 2-adaptive distinguisher for $\mathsf{F} \oplus \mathsf{G}$. This implies that there exists a non-negligible function $\epsilon(n)$ for n such that

$$|Pr[b_1 = 1] - Pr[b_2 = 1]| \geq \epsilon(n). \tag{13}$$

Since R^n is a uniformly random function from $\{0,1\}^n$ to $\{0,1\}^n$, for any x_1 and $x_2 \in \{0,1\}^n$, the distribution of $y_1 \leftarrow \mathsf{R}^n(x_1)$ and $y_2 \leftarrow \mathsf{R}^n(x_2)$ is equivalent to the uniform random of length n. In other words, the distribution of b_2 and b_3 are equivalent to one another:

$$|Pr[b_2 = 1] - Pr[b_3 = 1]| = 0. \tag{14}$$

Consider one more case described as follows.

$$\begin{aligned} &b_4 \leftarrow \mathcal{D}(y_1, y_2) \text{ where,} \\ &\quad k_1 \leftarrow \mathsf{Gen}_{\mathsf{G}}(1^n) \\ &\quad x_1 \leftarrow \mathcal{D}(1^n) \\ &\quad z_1 \leftarrow_{\$} \{0,1\}^n \\ &\quad y_1 \leftarrow z_1 \oplus \mathsf{G}_{k_1}(x_1) \\ &\quad x_2 \leftarrow \mathcal{D}(1^n) \\ &\quad z_2 \leftarrow_{\$} \{0,1\}^n \\ &\quad y_2 \leftarrow z_2 \oplus \mathsf{G}_{k_1}(x_2) \end{aligned}$$

For any x and key k in $\{0,1\}^n$, the distribution of y is uniform if $z \leftarrow_{\$} \{0,1\}^n$ and $y \leftarrow z \oplus \mathsf{G}_k(x)$. Thus, the distribution of y_1 and y_2 are uniform, which implies that the distribution of b_4 and b_3 are equal to each other. Hence, by (14),

$$|Pr[b_2 = 1] - Pr[b_3 = 1]| = |Pr[b_2 = 1] - Pr[b_4 = 1]| = 0. \tag{15}$$

Finally, we have

$$\begin{aligned}
Pr[b_{\mathsf{A}} = b_{\mathsf{B}}] &= Pr[b_{\mathsf{A}} = 1] \cdot Pr[b_{\mathsf{B}} = 1 : b_{\mathsf{A}} = 1] + Pr[b_{\mathsf{A}} = 0] \cdot Pr[b_{\mathsf{B}} = 0 : b_{\mathsf{A}} = 0] \\
&= \frac{1}{2} \cdot (1 - Pr[b_1 = 1]) + \frac{1}{2} \cdot Pr[b_4 = 1] \\
&= \frac{1}{2} \cdot (1 + (Pr[b_4 = 1] - Pr[b_1 = 1])) \\
&\leq \frac{1}{2} \cdot (1 + \epsilon(n)) \qquad \text{by (13) and (15)} \\
&= \frac{1}{2} + \frac{\epsilon(n)}{2}.
\end{aligned}$$

Therefore, the protocol Bit-Agreement(1^n) has non-negligible correlation in $\epsilon(n)$. □

Claim 4.2 (Security with overwhelming probability δ)**.** *The protocol Bit-Agreement*(1^n) *has overwhelming* δ*.*

Proof. We want to show that there exists an overwhelming function $\delta(n)$ such that for all efficient distinguishers $\mathcal{D} \in$ PPT and $\mathcal{D}$'s own randomness r,

$$\Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}} : b_{\mathsf{A}} = b_{\mathsf{B}}] \leq 1 - \frac{\delta(n)}{2}.$$

Since

$$\Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}}] = \Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}} : b_{\mathsf{A}} = b_{\mathsf{B}}] + \Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}} : b_{\mathsf{A}} \neq b_{\mathsf{B}}],$$

we have

$$\Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}} : b_{\mathsf{A}} = b_{\mathsf{B}}] \leq \Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}}].$$

Hence, it suffices to show that there exists an overwhelming $\delta(n)$ such that

$$\Pr[\mathcal{D}_r(z_1, y_1, z_2) \to b_{\mathsf{A}}] \leq 1 - \frac{\delta(n)}{2}.$$

Consider the following five cases, which define the distributions of the transcript triplet (z_1, y_1, z_2).

case 1	case 2	case 3	case 4	case 5
$k_1 \leftarrow \mathsf{Gen}_{\mathsf{F}}(1^n)$	$k_1 \leftarrow \mathsf{Gen}_{\mathsf{F}}(1^n)$	$x_1 \leftarrow \mathcal{D}(1^n)$		$k_2 \leftarrow \mathsf{Gen}_{\mathsf{G}}(1^n)$
$k_2 \leftarrow \mathsf{Gen}_{\mathsf{G}}(1^n)$		$z_1 \leftarrow \mathsf{R}^n(x_1)$	$x_1 \leftarrow \mathcal{D}(1^n)$	$x_1 \leftarrow \mathcal{D}(1^n)$
$x_1 \leftarrow \mathcal{D}(1^n)$	$x_1 \leftarrow \mathcal{D}(1^n)$	$y_1 \leftarrow z_1 \oplus \mathsf{R}^n(x_1)$	$z_1 \leftarrow_{\$} \{0,1\}^n$	$z_1 \leftarrow_{\$} \{0,1\}^n$
$z_1 \leftarrow \mathsf{F}_{k_1}(x_1)$	$z_1 \leftarrow \mathsf{F}_{k_1}(x_1)$	$x_2 \leftarrow \mathcal{D}(y_1)$	$y_1 \leftarrow z_1 \oplus \mathsf{R}^n(x_1)$	$y_1 \leftarrow z_1 \oplus \mathsf{G}_{k_2}(x_1)$
$y_1 \leftarrow z_1 \oplus \mathsf{G}_{k_2}(x_1)$	$y_1 \leftarrow z_1 \oplus \mathsf{R}^n(x_1)$	$z_2 \leftarrow \mathsf{R}^n(x_2)$	$x_2 \leftarrow \mathcal{D}(y_1)$	$x_2 \leftarrow \mathcal{D}(y_1)$
$x_2 \leftarrow \mathcal{D}(y_1)$	$x_2 \leftarrow \mathcal{D}(y_1)$		$z_2 \leftarrow_{\$} \{0,1\}^n$	$z_2 \leftarrow_{\$} \{0,1\}^n$
$z_2 \leftarrow \mathsf{F}_{k_1}(x_2)$	$z_2 \leftarrow \mathsf{F}_{k_1}(x_2)$			

We define $(z_1, y_1, z_2)_i$ to be the transcript triplet from the ith case. Then, ϵ_{ij} is defined as

$$|\Pr[\mathcal{D}(z_1, y_1, z_2)_i \to 1] - \Pr[\mathcal{D}(z_1, y_1, z_2)_j \to 1]| = \epsilon_{ij}.$$

By the non-adaptive security of G, ϵ_{12} is negligible. Also, ϵ_{23} is negligible due to the non-adaptive security of F. As we have seen in the proof of claim 4.1, case 3 is equivalent to case 4. Hence,

$\epsilon_{34} = 0$. The non-adaptive security of G implies that ϵ_{45} is negligible. Then we have, by triangle inequality,

$$|\Pr[\mathcal{D}(z_1, y_1, z_2)_5 \to 1] - \Pr[\mathcal{D}(z_1, y_1, z_2)_1]| \leq \sum_{i=1}^{4} \epsilon_{i(i+1)}$$

$$= \epsilon_{12} + \epsilon_{23} + \epsilon_{34} + \epsilon_{45} = \epsilon_{12} + \epsilon_{23} + \epsilon_{45} \stackrel{def}{=} \epsilon. \tag{16}$$

Since $\epsilon_{12}, \epsilon_{23}$, and ϵ_{45} are negligible, ϵ is negligible in n. We define δ to be $1 - \epsilon$. It is easy to see that δ is overwhelming in n since ϵ is negligible. Finally, we complete the proof of claim 4.2 as

$$\begin{aligned}
&\Pr[\mathcal{D}(z_1, y_1, z_2) \to b_{\mathsf{A}}] \\
&= \Pr[b_{\mathsf{A}} = 0] \cdot \Pr[\mathcal{D}(z_1, y_1, z_2) \to 0 : b_{\mathsf{A}} = 0] + \Pr[b_{\mathsf{A}} = 1] \cdot \Pr[\mathcal{D}(z_1, y_1, z_2) \to 1 : b_{\mathsf{A}} = 1] \\
&= \frac{1}{2} \cdot (\Pr[\mathcal{D}(z_1, y_1, z_2) \to 0 : b_{\mathsf{A}} = 0] + \Pr[\mathcal{D}(z_1, y_1, z_2) \to 1 : b_{\mathsf{A}} = 1]) \\
&= \frac{1}{2} \cdot (1 - \Pr[\mathcal{D}(z_1, y_1, z_2) \to 1 : b_{\mathsf{A}} = 0] + \Pr[\mathcal{D}(z_1, y_1, z_2) \to 1 : b_{\mathsf{A}} = 1]) \\
&= \frac{1}{2} \cdot (1 + (\Pr[\mathcal{D}(z_1, y_1, z_2)_5 \to 1] - \Pr[\mathcal{D}(z_1, y_1, z_2)_1])) \\
&\leq \frac{1}{2} \cdot (1 + \epsilon) \qquad \text{by (16)} \\
&= \frac{1}{2} \cdot (1 + (1 - \delta)) \\
&= 1 - \frac{\delta(n)}{2}.
\end{aligned}$$

□

Claim 4.3 (Uniform-transcript)**.** *The protocol Bit-Agreement*(1^n) *is a uniform-transcript bit agreement.*

Proof. Notice that the value of y_1 is related to the value of z_1 in the protocol, and as long as z_1 is uniformly random, y_1 is also uniformly random. Let R_i denote a uniformly random vector in $(\{0,1\}^n)^i$. Therefore, we want to show the following inequality for any PPT adversary (distinguisher) $\mathcal{A}$.

$$\left| \Pr_{b_A \leftarrow_\$ \{0,1\}}[\mathcal{A}_r(\mathsf{z_1}, \mathsf{y_1 z_2}) = 1] - \Pr_{b_A \leftarrow_\$ \{0,1\}}[\mathcal{A}_r(\mathsf{R}_3) = 1 : \mathsf{R}_3 \leftarrow_\$ (\{0,1\}^n)^3] \right| \leq \epsilon(n) \tag{17}$$

Then we define two games in which adversary $\mathcal{A}$ distinguishes a transcript from uniform random. Game $\mathcal{G}_0$ is for an adversary to distinguish (z_1, y_1, z_2) from R_3 while $b_{\mathsf{A}} = 0$, and game $\mathcal{G}_1$ is for an adversary to distinguish (z_1, y_1, z_2) from R_3 while $b_{\mathsf{A}} = 1$. We denote the advantage of adversary $\mathcal{A}$ in $\mathcal{G}_0$ and $\mathcal{G}_1$ as $\mathbf{Adv}_{\mathcal{G}_0}$ and $\mathbf{Adv}_{\mathcal{G}_1}$, respectively. Winning $\mathcal{G}_0$ (or $\mathcal{G}_1$) means that one distinguishes (z_1, y_1, z_2) from R_3 with a non-negligible advantage in $\mathcal{G}_0$ (or $\mathcal{G}_1$). Finally, suppose that there exists a PPT adversary $\mathcal{A}$ that wins either of the games.

In game $\mathcal{G}_1$, since both z_1 and z_2 are randomly chosen from $\{0,1\}^n$, distinguishing (z_1, y_1, z_2) from R_3 is equivalent to distinguishing $(r_1, G(r_2), r_3)$ from R_3. Hence, if $\mathcal{A}$ wins $\mathcal{G}_1$, it also distinguishes G from uniform random functions with non-negligible advantage. This is clearly impossible by the non-adaptive security of G.

Assume that $\mathcal{A}$ wins game $\mathcal{G}_0$ by distinguishing, with non-negligible probability,

$$Dist(0) : (z_1, y_1, z_2) = (\mathsf{F}(r), \mathsf{F}(r) \oplus \mathsf{G}(r), \mathsf{F}(\mathcal{D}(\mathsf{F}(r) \oplus \mathsf{G}(r))) \oplus \mathsf{D}(\mathsf{G}(\mathsf{F}(r) \oplus \mathsf{G}(r))))$$

from a uniform random triplet R_3 for $r \leftarrow_{\$} \{0,1\}^n$. Consider the following distributions.

$$Dist(1) : (\mathsf{F}(r_1), r_2, \mathsf{F}(\mathcal{D}(\mathsf{F}(r_1))) \oplus \mathsf{G}(\mathcal{D}(\mathsf{F}(r_1))))$$

for r_1 and $r_2 \leftarrow_{\$} \{0,1\}^n$.

$$Dist(2) : (r_1, r_2, \mathsf{F}(\mathcal{D}(r_1 \oplus r_2)) \oplus \mathsf{G}(\mathcal{D}(r_1 \oplus r_2)))$$

for r_1 and $r_2 \leftarrow_{\$} \{0,1\}^n$.

$$Dist(3) : (r_1, r_2, r_3 \oplus r_4) = (r_1, r_2, r_5) = \mathsf{R}_3$$

where $r_5 = r_3 \oplus r_4$ for r_1, r_2, r_3 and $r_4 \leftarrow_{\$} \{0,1\}^n$.

Since $\mathcal{A}$ distinguishes Dist(0) from Dist(3) with non-negligible advantage, $\mathcal{A}$ distinguishes Dist(i) from Dist(j) for $i \neq j$ with non-negligible probability. This is clearly a contradiction since the above distributions only negligibly deviate from each other by the non-adaptive security of both F and G.

Therefore, the advantage of $\mathcal{A}$ to distinguish the transcript from R_3 is $\mathbf{Adv}_{\mathcal{G}_0}/2 + \mathbf{Adv}_{\mathcal{G}_1}/2$ which is negligible since both $\mathbf{Adv}_{\mathcal{G}_0}$ and $\mathbf{Adv}_{\mathcal{G}_1}$ are negligible. This directly validates the inequality (17). □

With Claims 4.1 and 4.2, we showed that the protocol Bit-Agreement(1^n) has non-negligible correlation ϵ and is secure with overwhelming probability δ. Also, we show the indistinguishability of messages from randoms with Claim 4.3. Thus, this implies the existence of a uniform-transcript key agreement. We eventually show that the $(k-1)$-adaptively secure parallel composition implies the existence of the $(2k-1)$-pass uniform-transcript key agreement for the case $k = 2$ with respect to a 2-adaptive distinguisher. We will generalize the same technique to the arbitrary k.

We just showed that the $(k-1)$-adaptively secure parallel composition implies the $(2k-1)$-pass uniform-transcript bit agreement for the case $k = 2$. However, notice that the 2-adaptive distinguisher $\mathcal{D}$, which we use to build Bit-Agreement(1^n), is not a general 2-adaptive distinguisher since $\mathcal{D}$ makes two adaptive queries. However, it is easy to see that we can construct the same 3-pass bit agreement protocol based on a general 2-adaptive distinguisher denoted by $\mathcal{D}_q$, where q is the any polynomial size of blocks queried by the distinguisher. Then, we build the same 3-pass uniform-transcript bit-agreement by replacing $x_1, x_2, y_1, y_2, z_1, z_2$ with $X_1, X_2, Y_1, Y_2, Z_1, Z_2$ in the Bit-Agreement(1^n), where X_i, Y_i, Z_i are q tuples. That is, $x_i = (x_{1i}, x_{2i}, \cdots, x_{qi})$ for i. Now we are ready to generalize the construction of Bit-Agreement(1^n) to the arbitrary $k \geq 2$. Denote $\mathcal{D}_q^k$ as a k-adaptive distinguisher with the query block of size q. Then, X_i, Y_i, Z_i are defined by q-tuples for all i as before. We provide the construction in Protocol 3. Obviously, we can extend the arguments of Claim 4.1, 4.2 and 4.3 to the $(2k-1)$-pass uniform transcript bit-agreement. To prove that the above bit agreement is secure with overwhelming probability, only the number of intermediate cases between the distributions of transcripts is increased according to the increased number of rounds. This completes the proof of Theorem 7. □

4.1.2 Intuitions of Adaptively Insecure Parallel Composition of F and G under UTKA

A γ-round uniform-transcript key agreement protocol (γ-UTKA), denoted by $\Phi_u^\gamma = (\mathsf{A}, \mathsf{B})$, is a uniform-transcript key agreement protocol consisting of two sub-protocols A and B, in which Alice

Protocol Bit-Agreement(1^n)

Alice	Transcript	Bob
$b_{\mathsf{A}} \leftarrow_{\$} \{0,1\}^n$		
$k_{\mathsf{A}} \leftarrow \mathsf{Gen}_{\mathsf{F}}(1^n)$		$k_{\mathsf{B}} \leftarrow \mathsf{Gen}_{\mathsf{G}}(1^n)$
FOR $i = 1$ to $k-1$ DO		
$X_i \leftarrow \mathcal{D}_q^k(Y_1, Y_2, \cdots, Y_{i-1})$		
If $b_{\mathsf{A}} = 0$,		
then $Z_i \leftarrow \mathsf{F}_{k_{\mathsf{A}}}(X_i)$		$X_i \leftarrow \mathcal{D}_q^k(Y_1, Y_2, \cdots, Y_{i-1})$
else $Z_i \leftarrow_{\$} (\{0,1\}^n)^q$	$\xrightarrow{Z_i}$	
	$\xleftarrow{Y_i}$	$Y_i \leftarrow Z_i \oplus \mathsf{G}_{k_{\mathsf{B}}}(X_i)$
ENDFOR		
$X_k \leftarrow \mathcal{D}_q^k(Y_1, Y_2, \cdots, Y_{k-1})$		
If $b_{\mathsf{A}} = 0$,		
then $Z_k \leftarrow \mathsf{F}_{k_{\mathsf{A}}}(X_k)$		$X_k \leftarrow \mathcal{D}_q^k(Y_1, Y_2, \cdots, Y_{k-1})$
else $Z_k \leftarrow_{\$} (\{0,1\}^n)^q$	$\xrightarrow{Z_k}$	$Y_k \leftarrow Z_k \oplus \mathsf{G}_{k_{\mathsf{B}}}(X_k)$
		$b_{\mathsf{B}} \leftarrow \mathcal{D}_q^k(Y_1, Y_2, \cdots, Y_k)$

Protocol 3: $(2k-1)$-pass uniform-transcript bit agreement based on a general k-adaptive distinguisher

(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.

We first provide the intuitive description of the parallel version of γ-UTKA in Protocol 4, which we will use to construct counter-example functions. Notice that Alice and Bob are *symmetric* to each other in Protocol 4. In particular, Bob's first message is completely independent of Alice's first message and is only dependent on his own private randomness.[2]

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 $\Phi_u = (\mathsf{A}, \mathsf{B})$, γ-UTKA described in Protocol 2. α_i and β_i denote the ith message computed by A and B respectively. We are give 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 a specially related pseudo-random strings with respect to the shared secret key. As we input the specially related pseudo-random strings to the parallel 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 $Q_1 = (0^n, 0^n, \cdots, 0^n)$ as:

[2]The main reason for using this parallel version of γ-UTKA is that it is easier 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. It is possible to construct the counter-example functions to show the same composition insecurity result by using $\gamma - UTKA$ 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.

Alice	Transcript	Bob
$r_{\mathsf{A}} \leftarrow_{\$} \{0,1\}^n$		$r_{\mathsf{B}} \leftarrow_{\$} \{0,1\}^n$
$\alpha_1 \leftarrow \mathsf{A}_1(r_{\mathsf{A}})$		$\beta_1 \leftarrow \mathsf{B}_1(r_{\mathsf{B}})$
	$\xrightarrow{\alpha_1}$	
	$\xleftarrow{\beta_1}$	
$\alpha_2 \leftarrow \mathsf{A}_2(r_{\mathsf{A}}, \beta_1)$		$\beta_2 \leftarrow \mathsf{B}_2(r_{\mathsf{B}}, \alpha_1)$
	$\xrightarrow{\alpha_2}$	
	$\xleftarrow{\beta_2}$	
	$\vdots$	
$\alpha_\gamma \leftarrow \mathsf{A}_\gamma(r_{\mathsf{A}}, \beta_1, \cdots, \beta_{\gamma-1})$		$\beta_\gamma \leftarrow \mathsf{B}_\gamma(r_{\mathsf{B}}, \alpha_1, \cdots, \alpha_{\gamma-1})$
	$\xrightarrow{\alpha_\gamma}$	
	$\xleftarrow{\beta_\gamma}$	
secret key $sk \leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{A}}, \beta_1, \cdots, \beta_\gamma)$		secret key $sk \leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{B}}, \alpha_1, \cdots, \alpha_\gamma)$

Protocol 4: Parallel version of γ-UTKA

- 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}, $\cdots$, $s_{(\gamma+2)1}$).

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

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

Inductively, for $2 \leq i \leq \gamma$, 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,

$$\begin{array}{ll} \beta_1 \leftarrow (\alpha_1 \oplus u_1) & \alpha_2 \leftarrow \mathsf{A}_2(r_{\mathsf{F}}, \beta_1) \\ \beta_2 \leftarrow (\alpha_2 \oplus u_2) & \alpha_3 \leftarrow \mathsf{A}_3(r_{\mathsf{F}}, \beta_1, \beta_2) \\ \vdots & \vdots \\ \beta_{i-1} \leftarrow (\alpha_{i-1} \oplus u_{i-1}) & \alpha_i \leftarrow \mathsf{A}_i(r_{\mathsf{F}}, \beta_1, \beta_2, \ldots, \beta_{i-1}) \end{array}$$

  Finally, F outputs (α_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. That is, we have the description G on Q_i by replacing all of F, r, A, and α in the above description with G, s, B, and β. Hence, G outputs (β_i, s_{2i}, $\cdots$, $s_{(\gamma+2)i}$) where s_{2i}, $\cdots$, $s_{(\gamma+2)i}$ are generated by $\mathsf{P}_{k_{\mathsf{G}}}(Q_i)$.

On Q_i for $2 \leq i \leq \gamma$, we demonstrate the individual outputs of F and G and the output of their parallel composition below.

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

Hence, we obtain $\alpha_\gamma \oplus \beta_\gamma$ by feeding the parallel composition of F and G with Q_γ to be $(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_{\gamma-1} \oplus \beta_{\gamma-1}, 0^n, 0^n)$.

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 $\mathsf{Enc}_{sk}(r_{1(\gamma+1)})$. Finally, 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 $Q_{\gamma+1}$.

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

$$\rightarrow (\mathsf{Enc}_{sk}(r_{1(\gamma+1)}) \oplus \mathsf{Enc}_{sk}(s_{1(\gamma+1)}), r_{1(\gamma+1)} \oplus s_{1(\gamma+1)}, r_{3(\gamma+1)} \oplus s_{3(\gamma+1)}, \cdots, r_{(\gamma+2)(\gamma+1)} \oplus s_{(\gamma+2)(\gamma+1)})$$

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 $(k_\mathsf{F}, 0^n, 0^n, \cdots, 0^n)$.
- 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.

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

4.1.3 Formal Construction of Non-Adaptively Secure Functions F and G

The underlying primitives used for the construction of F and G are two PRPs, $\tilde{\pi} : K \times (\{0,1\}^n)^{\gamma+2} \rightarrow (\{0,1\}^n)^{\gamma+2}$ and $\pi : K \times \{0,1\}^n \rightarrow \{0,1\}^n$, where K is key space $\{0,1\}^n$. Also, we are given a black-box access to the parallel version of γ-UTKA denoted by $\Phi_u = (\mathsf{A}, \mathsf{B})$ described in Section 4.1.2.

Construction of F

1. If $I = (u_1 = 0^n, u_2 = 0^n, \cdots, u_{\gamma+2} = 0^n)$, then
   Output $(v_1, v_2, \cdots, v_{\gamma+2})$ where
   $(r_1, r_2, \cdots, r_{\gamma+2}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(0^n, 0^n, \cdots, 0^n)$
   $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (\mathsf{A}_1(r_1), r_2, \cdots, r_{\gamma+2})$

2. If $I = (u_1 \neq 0^n, \cdots, u_i \neq 0^n, u_{i+1} = 0^n, \cdots, u_{\gamma+2} = 0^n)$, then
   Output $(v_1, v_2, \cdots, v_{\gamma+2})$ where
   $(a_1, a_2, \cdots, a_{\gamma+2}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(0^n, 0^n, \cdots, 0^n)$
   $(r_1, r_2, \cdots, r_{\gamma+2}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \cdots, u_{\gamma+2})$
   $\alpha_1 \leftarrow \mathsf{A}_1(a_1)$
   For $j = 1$ to γ Do
      $\beta_j \leftarrow \alpha_j \oplus u_j$
      $\alpha_{j+1} \leftarrow \mathsf{A}_{j+1}(a_1, \beta_1, \cdots, \beta_j)$
   END For
   $sk_{\mathsf{F}} \leftarrow \alpha_{j+1}$

   (a) If $i < \gamma$, then
      $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (\alpha_{i+1}, r_2, \cdots, r_{\gamma+2})$

   (b) Else If $i = \gamma$, then
      $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (\pi_{sk_{\mathsf{F}}}(r_1), r_1, r_2, \cdots, r_{\gamma+1})$

   (c) Else If $i = \gamma + 2$, then
      $(b_1, b_2, \cdots, b_{\gamma+2}) \leftarrow \mathsf{F}(u_1, u_2, \cdots, u_\gamma, 0^n, 0^n)$
      i. If $\pi^{-1}_{sk_{\mathsf{F}}}(b_1 \oplus u_{\gamma+1}) = b_2 \oplus u_{\gamma+2}$, then
         $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (k_{\mathsf{F}}, 0^n, \cdots, 0^n)$
      ii. else $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (r_1, r_2, \cdots, r_{\gamma+2})$

   (d) Else $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow (r_1, r_2, \cdots, r_{\gamma+2})$

3. If the input is not any of above cases, then
   $(v_1, v_2, \cdots, v_{\gamma+2}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_1, u_2, \cdots, u_{\gamma+2})$

Algorithm 5: The algorithm of function F

We formally define our non-adaptively secure pseudo-random function F to map from $(\{0,1\}^n)^{\gamma+2}$ to $(\{0,1\}^n)^{\gamma+2}$. Furthermore, F internally possesses a secret key $k_{\mathsf{F}} \in K$. The formal definition of function F is provided in Algorithm 5.

The construction of G is symmetric to F. That is, replacing F, A, r, and $(k_{\mathsf{F}}, 0^n, \cdots, 0^n)$ in Algorithm 5 with G, B, s, and $(0^n, k_{\mathsf{G}}, 0^n, \cdots, 0^n)$ respectively will provide us with the formal construction of G.

Claim 4.4. *The functions F and G are secure against any non-adaptive PPT adversary $\mathcal{A}(q,t)$, running in time t and making at most q non-adaptive queries, where t and q are any polynomials of security parameter n.*

Proof. By hybrid argument, we reduce the security of function F to the indistinguishability of the underlying PRPs and the γ-UTKA. Let $\mathcal{A}$ be a PPT adversary making q queries and running for time t. Notice that F and G are structurally identical to one another. Hence, it suffices to show that F is non-adaptively secure. To prove the claim, we will show the following inequality.

$$\mathbf{Adv}^{\mathsf{F}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\mathsf{A}}_{\mathcal{A}} + \frac{q}{2^n}.$$

First, assume that to obtain the secret key k_{F}, $\mathcal{A}$ makes q queries of the form in $(u_1 \neq 0^n, \cdots, u_{\gamma+2} \neq 0^n)$ evoking case 2 of Algorithm 5. Then, let $\mathcal{A}$ fix the first γ queries to be the same through the q queries. This implies that sk_{F}, b_1, and b_2 are fixed in the condition $\pi^{-1}_{sk_{\mathsf{F}}}(b_1 \oplus u_{\gamma+1}) = b_2 \oplus u_{\gamma+2}$ in the case 2(c) of Algorithm 5. This only helps the adversary $\mathcal{A}$ by allowing it to choose $u_{\gamma+1}$ and $u_{\gamma+2}$ to satisfy the condition. Since π is a permutation, there uniquely exists $u_{\gamma+1}$ for each $u_{\gamma+2}$, which satisfies the condition. Hence,

$$\Pr[\pi^{-1}_{k'}(\alpha \oplus u_{\gamma+1}) = r_1 \oplus u_{\gamma+2} : u_{\gamma+1}, u_{\gamma+2} \leftarrow_{\$} \{0,1\}^n] \leq \frac{1}{2^n}. \tag{18}$$

Since $\mathcal{A}$ makes q queries, the probability of successfully finding a pair of such $u_{\gamma+1}$ and $u_{\gamma+2}$ is $q/2^n$, which constitutes the final term of the inequality (18).

Consider that $\mathcal{A}$ makes queries that fall into case 2(c), 2(d), and 3 of Algorithm 5. As we showed above, q non-adaptive queries can satisfy the condition in 2(c) with only negligible probability. Thus, assume that all the queries made here do not satisfy the condition. Hence, F outputs, upon q queries,

$$(r_{11}, r_{21}, \cdots, r_{(\gamma+2)1}), (r_{12}, r_{22}, \cdots, r_{(\gamma+2)2}), \cdots, (r_{1q}, r_{2q}, \cdots, r_{(\gamma+2)q}), \tag{19}$$

where r_{ij} is the ith coordinate of the jth query for $1 \leq i \leq \gamma+2$ and $1 \leq j \leq q$.

Proceeding by hybrid argument, assume that $\mathcal{A}$ distinguishes (19) from uniform distributions with a non-negligible probability ξ,

$$(r^*_{11}, r^*_{21}, \cdots, r^*_{(\gamma+2)1}), (r^*_{12}, r^*_{22}, \cdots, r^*_{(\gamma+2)2}), \cdots, (r^*_{1q}, r^*_{2q}, \cdots, r^*_{(\gamma+2)q}), \tag{20}$$

where $(r^*_{1i}, r^*_{2i}, \cdots, r^*_{(\gamma+2)i}) \leftarrow_{\$} (\{0,1\}^n)^{\gamma+2}$ for $1 \leq i \leq q$.

Let H_i be the hybrid distribution as

$$H_i = h_1, \cdots, h_i, h^*_{i+1}, \cdots, h^*_q$$

for $h_i = (r_{1i}, \cdots, r_{(\gamma+2)i})$ from (19) and $h^*_j = (r^*_{1j}, \cdots, r^*_{(\gamma+2)j})$ from (20).

Since $H_q - H_0 \geq \xi$, there exists i such that $|H_j - H_{j-1}| \geq \frac{\xi}{q}$. Consider the ith hybrid,

$$H_i(x) = h_1, h_2, \cdots, h_{i-1}, x, h^*_{i+1}, \cdots, h^*_q.$$

Then we can build a PPT distinguisher $\mathcal{D}$ using $\mathcal{A}$ as a sub-routine. With an unknown distribution u from $(\{0,1\})^{\gamma+2}$, $\mathcal{D}$ constructs and queries $H_i(u)$ to $\mathcal{A}$. Since $\mathcal{A}$ distinguishes F from a uniform random function with the probability ξ, $\mathcal{D}$ will distinguish $\tilde{\pi}$ from a uniform random function with the probability $\frac{\xi}{q}$, which is non-negligible.

Consider that $\mathcal{A}$ makes q queries in the form of $Q_i = (u_{1i} \neq 0^n, u_{2i} \neq 0^n, \cdots, u_{\gamma i} \neq 0^n, 0^n, 0^n)$, that provoke case 2(b) of F in Algorithm 5. Then, the corresponding outputs can be written as,

$$(\pi_{k_1}(r_{11}), r_{11}, r_{31}, \cdots, r_{(\gamma+2)1}), (\pi_{k_2}(r_{12}), r_{12}, r_{32}, \cdots, r_{(\gamma+2)2}), \cdots, (\pi_{k_q}(r_{1q}), r_{1q}, r_{3q}, \cdots, r_{(\gamma+2)q}),$$

where $(r_{1i}, r_{2i}, \cdots, r_{(\gamma+2)i}) \leftarrow \tilde{\pi}_{k_\mathsf{F}}(Q_i)$ for $1 \leq i \leq q$ and k_i is computed by $\mathsf{A}_{\gamma+1}(r, \mathsf{Trans}^{\mathsf{A}})$ for unknown r. distinguishing the above distribution from uniform random implies that one of the following two cases must be true. First, the adversary $\mathcal{A}$ distinguishes the last $\gamma+1$ coordinates (all coordinates except for the first coordinate) of the outputs from uniform random. Second, the first two coordinates from uniform random. Assume that the first case is true. Then, $\mathcal{A}$ can distinguish all the output of F upon queries of case 2(c)ii, 2(d) or 3 by simply ignoring the second coordinate of the outputs. This is clearly a contradiction to the non-adaptive security of F in those cases proven above. Assume that the second case holds. Distinguishing the first two coordinates of the outputs from uniform random is equivalent to distinguishing,

$$(\pi(r_1), r_1), (\pi(r_2), r_2), \cdots, (\pi(r_q), r_q), \tag{21}$$

where $r_i \leftarrow_\$ \{0,1\}^n$ for $1 \leq i \leq q$, from the uniform distribution,

$$(a_1, b_1), (a_2, b_2), \cdots, (a_q, b_q), \tag{22}$$

where a_i and $b_i \leftarrow_\$ \{0,1\}^n$ for $1 \leq i \leq q$.

Suppose that $\mathcal{A}$ distinguishes (21) from (22) with a non-negligible probability ξ. Define a hybrid distribution H_i as

$$H_i = (\pi(r_1), r_1), \cdots, (\pi(r_i), r_i), (\pi(r_{i+1}), b_{i+1}), \cdots, (\pi(r_q), b_q), \tag{23}$$

where r_i and $b_i \leftarrow_\$ \{0,1\}^n$ for $1 \leq i \leq q$.

Since $\mathcal{A}$ distinguishes (21) from (22) with probability ξ, $|H_0 - H_q| \geq \xi$, there exists i s.t. $|H_i - H_{i+1}| \geq \frac{\xi}{q}$. Hence, let $H_i(x)$ for $x \in (\{0,1\}^n)^2$ be

$$H_i(x) = (\pi(r_1), r_1), \cdots, (\pi(r_{i-1}), r_{i-1}), x, (\pi(r_{i+1}), b_{i+1}), \cdots, (\pi(r_q), b_q).$$

Upon an unknown distribution $(\alpha, \beta) \in (\{0,1\}^n)^2$, we can build a distinguisher $\mathcal{D}$, which determines whether (α, β) comes from (21) or from (22) with a non-negligible probability $\frac{\xi}{q}$ as $\mathcal{D}$ queries $H_i((\alpha, \beta))$ to $\mathcal{A}$. Therefore, $\mathcal{D}$ distinguishes π from a uniform random function with a non-negligible probability.

Consider that $\mathcal{A}$ makes q queries in the form of $(u_1 \neq 0^n, \cdots, u_i \neq 0^n, u_{i+1} = 0^n, \cdots, u_{\gamma+2} = 0^n)$ for $i < \gamma$. That is, all the q queries fall into case 1 or 2(a) of Algorithm 5. By replacing π in the above hybrid argument with A, we can also show that if $\mathcal{A}$ breaks the indistinguishability of F, then we can construct a PPT distinguisher $\mathcal{D}$, which breaks the indistinguishability of messages. □

4.1.4 Adaptive Insecurity of Parallel Composition of F and G

Claim 4.5. *The parallel composition $\mathsf{F} \oplus \mathsf{G}$ is breakable by $\gamma+2$ adaptive queries.*

Proof. Let r_{ij} denote the ith randomness upon the jth input. To initiate the key agreement, our first special input is $(0^n, 0^n, \cdots, 0^n)$. Then, F gets into case 1 of Algorithm 5 and computes its first message α_1 and outputs $(\alpha_1, r_{21}, \cdots, r_{(\gamma+2)1})$. So does G. Hence, the output of the parallel composition on the first input query is

$$(\alpha_1 \oplus \beta_1, r_{21} \oplus s_{21}, \cdots, r_{(\gamma+2)1} \oplus s_{(\gamma+2)1}).$$

Our second adaptive query is $Q_2 = (\alpha_1 \oplus \beta_1, 0^n, \cdots, 0^n)$, where $\alpha_1 \oplus \beta_1$ comes from the output of the parallel composition on Q_1. Q_2 evokes the case 2 of Algorithm 5 unless $\alpha_1 \oplus \beta_1 = 0^n$. The only case that $\alpha_1 \oplus \beta_1 = 0^n$ is $\alpha_1 = \beta_1$ which occurs with negligible probability. By the computations of "For" loop of case 2, F obtains α_1, α_2, $\cdots$, $\alpha_{\gamma+1}$ as follows.

	Before entering For loop	
$\alpha_1 \leftarrow \mathsf{A}_1(a_1)$		
$\beta_1 \leftarrow \alpha_1 \oplus u_1 = \alpha_1 \oplus (\alpha_1 \oplus \beta_1)$	$\alpha_2 \leftarrow \mathsf{A}_2(a_1, \beta_1)$	$j = 1$
$\beta_2 \leftarrow \alpha_2 \oplus u_2 = \alpha_2 \oplus 0^n$	$\alpha_3 \leftarrow \mathsf{A}_3(a_1, \beta_1, \alpha_2)$	$j = 2$
$\beta_3 \leftarrow \alpha_3 \oplus u_3 = \alpha_3 \oplus 0^n$	$\alpha_4 \leftarrow \mathsf{A}_4(a_1, \beta_1, \alpha_2, \alpha_3)$	$j = 3$
$\vdots$	$\vdots$	$\vdots$
$\beta_\gamma \leftarrow \alpha_\gamma \oplus u_\gamma = \alpha_\gamma \oplus 0^n$	$\alpha_{\gamma+1} \leftarrow \mathsf{A}_{\gamma+1}(a_1, \beta_1, \alpha_2, \cdots, \alpha_\gamma)$	$j = \gamma$

Since the first coordinate is the only non-zero coordinate, Q_2 evokes case (a) of case 2 after the For loop. Hence, function F outputs $(\alpha_1, r_{22}, r_{32}, \cdots, r_{(\gamma+2)2})$ where $(r_{12}, r_{22}, \cdots, r_{(\gamma+2)2})$ is generated by $\tilde{\pi}_{k_\mathsf{F}}(Q_2)$. G undertakes the identical course of computation, so it outputs $(\beta_2, , s_{22}, \cdots, s_{(\gamma+2)2})$.

Inductively, for $2 \leq i \leq \gamma$, our ith adaptive query to the parallel composition is defined as

$$Q_i = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_{i-1} \oplus \beta_{i-1}, 0^n, \cdots, 0^n) \tag{24}$$

where $\alpha_l \oplus \beta_l$ is obtained from the output of F on Q_l. Then, the output of the parallel composition on Q_i is

$$(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_i \oplus \beta_i, r_{(i+1)i} \oplus s_{(i+1)i}, \cdots, r_{(\gamma+2)i} \oplus s_{(\gamma+2)i}) \tag{25}$$

where $(r_{1i}, \cdots, r_{(\gamma+2)i})$ is generated by $\tilde{\pi}_{k_\mathsf{F}}(Q_i)$ and $(s_{1i}, \ldots, s_{(\gamma+2)i})$ is generated by $\tilde{\pi}_{k_\mathsf{G}}(Q_i)$.

Hence, the γth adaptive query (the final case of the above inductive cases) is $Q_\gamma = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_{\gamma-1} \oplus \beta_{\gamma-1}, 0^n, 0^n, 0^n)$ by (24), and the corresponding output is $(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, r_{(\gamma+1)i} \oplus s_{(\gamma+1)i}, r_{(\gamma+2)i} \oplus s_{(\gamma+2)i})$ by (25).

Now, we define our $(\gamma+1)$th adaptive query as $Q_{\gamma+1} = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, 0^n, 0^n)$ which evokes case 2. Again, F retrives a_1 by computing $\pi_{k_\mathsf{F}}(Q_1)$. F performs the computations of For loop as follows.

	Before entering For loop	
$\alpha_1 \leftarrow \mathsf{A}_1(a_1)$		
$\beta_1 \leftarrow \alpha_1 \oplus u_1 = \alpha_1 \oplus (\alpha_1 \oplus \beta_1)$	$\alpha_2 \leftarrow \mathsf{A}_2(a_1, \beta_1)$	$j = 1$
$\beta_2 \leftarrow \alpha_2 \oplus u_2 = \alpha_2 \oplus (\alpha_2 \oplus \beta_2)$	$\alpha_3 \leftarrow \mathsf{A}_3(a_1, \beta_1, \beta_2)$	$j = 2$
$\beta_3 \leftarrow \alpha_3 \oplus u_3 = \alpha_3 \oplus (\alpha_3 \oplus \beta_3)$	$\alpha_4 \leftarrow \mathsf{A}_4(a_1, \beta_1, \beta_2, \beta_3)$	$j = 3$
$\vdots$	$\vdots$	$\vdots$
$\beta_\gamma \leftarrow \alpha_\gamma \oplus u_\gamma = \alpha_\gamma \oplus (\alpha_\gamma \oplus \beta_\gamma)$	$\alpha_{\gamma+1} \leftarrow \mathsf{A}_{\gamma+1}(a_1, \beta_1, \beta_2, \cdots, \beta_\gamma)$	$j = \gamma$

Since $Q_{\gamma+1}$'s first γ coordinates are adaptively generated from the previous adaptive queries and non-zero with overwhelming probability, $\alpha_{\gamma+1}$ is a properly computed shared key, sk_F and $Q_{\gamma+1}$

evokes case 2(b). Thus, F outputs $(\pi_{sk_\mathsf{F}}(r_{1(\gamma+1)}), r_{1(\gamma+1)}, r_{2(\gamma+1)}, \cdots, r_{(\gamma+1)(\gamma+1)})$ where $(r_{1(\gamma+1)}, \cdots, r_{(\gamma+2)(\gamma+1)})$ is generated from $\tilde{\pi}_{k_\mathsf{F}}(Q_{\gamma+1})$. On $Q_{\gamma+1}$, G also performs the identical course of computations, so it outputs $(\pi_{sk_\mathsf{G}}(s_{1(\gamma+1)}), s_{1(\gamma+1)}, s_{2(\gamma+1)}, \cdots, s_{(\gamma+1)(\gamma+1)})$. Hence, the output of the parallel composition on $Q_{\gamma+1}$ is

$$(\pi_{sk_\mathsf{F}}(r_{1(\gamma+1)}) \oplus \pi_{sk_\mathsf{G}}(s_{1(\gamma+1)}), r_{1(\gamma+1)} \oplus s_{1(\gamma+1)}, r_{2(\gamma+1)} \oplus s_{2(\gamma+1)}, \cdots, r_{(\gamma+1)(\gamma+1)} \oplus s_{(\gamma+1)(\gamma+1)})$$

The final $(\gamma+2)$th adaptive query is defined as

$$Q_{\gamma+2} = (\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, \underbrace{\pi_{sk_\mathsf{F}}(r_{1(\gamma+1)}) \oplus \pi_{sk_\mathsf{G}}(s_{1(\gamma+1)})}_{u_{\gamma+1}}, \underbrace{r_{1(\gamma+1)} \oplus s_{1(\gamma+1)}}_{u_{\gamma+2}}). \tag{26}$$

On $Q_{\gamma+2}$, F obtains the shared key sk_F by simulating the previous round computation with the first γ coordinates of $Q_{\gamma+2}$. $Q_{\gamma+2}$ evokes case 2(c) of Algorithm 5 as the final two coordinates of $Q_{\gamma+2}$ are non-zero with overwhelming probability. Now F computes b_1 and b_2 by simulating the output of the previous round of itself such that $(b_1, b_2, \cdots, b_2)$ is the output from $\mathsf{F}(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \cdots, \alpha_\gamma \oplus \beta_\gamma, 0^n, 0^n) = \mathsf{F}(Q_{\gamma+1})$. Therefore,

$$b_1 = \pi_{k_{sk_\mathsf{F}}}(r_{1(\gamma+1)}) \tag{27}$$

$$b_2 = r_{1(\gamma+1)}. \tag{28}$$

Then, F verifies that equality $\pi^{-1}_{sk_\mathsf{F}}(b_1 \oplus u_{\gamma+1}) = b_2 \oplus u_{\gamma+2}$ holds as

$$\begin{aligned}
\pi^{-1}_{sk_\mathsf{F}}(b_1 \oplus u_{\gamma+1}) &= \pi^{-1}_{sk_\mathsf{F}}(\pi_{k_{sk_\mathsf{F}}}(r_{1(\gamma+1)}) \oplus \pi_{sk_\mathsf{F}}(r_{1(\gamma+1)}) \oplus \pi_{sk_\mathsf{G}}(s_{1(\gamma+1)})) && \text{by (27)and (26)} \\
&= \pi^{-1}_{sk_\mathsf{F}}(\pi_{sk_\mathsf{G}}(s_{1(\gamma+1)})) \\
&= s_{1(\gamma+1)} && \text{since } sk_\mathsf{F} = sk_\mathsf{G} \\
&= (r_{1(\gamma+1)} \oplus r_{1(\gamma+1)}) \oplus s_{1(\gamma+1)} \\
&= r_{1(\gamma+1)} \oplus (r_{1(\gamma+1)} \oplus s_{1(\gamma+1)}) \\
&= b_2 \oplus u_{\gamma+2} && \text{by (28)and (26)}
\end{aligned}$$

Therefore, F outputs $(k_\mathsf{F}, 0^n, \cdots, 0^n)$. Similarly, G performs the same course of computation as does F, so it outputs $(0^n, k_\mathsf{G}, 0^n, \cdots, 0^n)$. Therefore, the final output of the parallel composition is $(k_\mathsf{F}, k_\mathsf{G}, 0^n, \cdots, 0^n)$ which reveals all the secret keys of F and G. □

Now, by having proved Claim 4.4 and 4.5, we obtain the following theorem of the impossibility of adaptively secure parallel composition under the existence of γ-UTKA.

Theorem 8. *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 $(\gamma+2)$-adaptive query breakable.*

Theorem 7 and 8 immediately substantiate the equivalence between the existence of UTKA and the above impossibility result as formally stated below.

Theorem 9. *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

4.2.1 Constructing UTKA from the Adaptive Insecurity of $\mathsf{G}(\mathsf{F}(\cdot))$

In this section, we briefly review the main idea behind Pietrzak's construction of key agreement, which is achieved from two pseudo-random functions whose sequential composition is not adaptively secure. Let us have two secure functions F and G such that their sequential composition $\mathsf{G}(\mathsf{F}(\cdot))$ is broken by a distinguisher $\mathcal{D}$ with k-adaptive queries. Alice chooses her secret bit b in which Bob will eventually know what bit b is. If $b = 1$, Alice obtains the first (starting) adaptive input from $\mathcal{D}$, and sends Bob the cypher-text c of the first input, encrypted by F. Bob simply computes $\mathsf{G}(c)$ and sends it back to Alice. Then, Alice again uses $\mathcal{D}$ to get an (the second) adaptive input based on $\mathsf{G}(c)$ and repeats the same process. If $b = 0$, then Alice simply chooses and sends a uniformly random string to Bob (every time) and Bob follows the same procedure as described above. As this process is performed repeatedly k times, at the end Bob obtains k adaptive outputs on which $\mathcal{D}$ outputs a bit consistent with Alice's secret bit with non-negligible probability. Hence, Bob obtains b with non-negligible probability. The pictorial description of the protocol is provided in Protocol 5. In fact, his bit agreement protocol satisfies the property of uniform-transcript. We prove this as

Pietrzak's Protocol Bit-Agreement(1^n)

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

Protocol 5: [Pie06] 3-pass bit agreement based on 2-adaptive distinguisher $\mathcal{D}$

a separate claim and formally restate Pietrzak's theorem below.

Claim 4.6. *The protocol Bit-Agreement(1^n) in [Pie06] is a uniform transcript bit agreement.*

Proof. To show that the protocol Bit-Agreement(1^n) has uniform transcript, we prove that for any PPT adversary $\mathcal{A}$,

$$\left| \Pr_{b_A \leftarrow_{\$} \{0,1\}} [\mathcal{A}_r(\mathsf{z_1}, \mathsf{y_1 z_2}) = 1] - \Pr_{b_A \leftarrow_{\$} \{0,1\}} [\mathcal{A}_r(\mathsf{R}_3) = 1 : \mathsf{R}_3 \leftarrow_{\$} (\{0,1\}^n)^3] \right| \leq \epsilon(n), \tag{29}$$

where $\epsilon(n)$ is a negligible function in n. First, we define two games for adversary $\mathcal{A}$ to distinguish transcript from random string. Game $\mathcal{G}_0$ is for the adversary to distinguish (z_1, y_1, z_2) from R_3 when $b_{\mathsf{A}} = 0$, and game $\mathcal{G}_1$ is for the adversary to distinguish (z_1, y_1, z_2) from R_3 when $b_{\mathsf{A}} = 1$. Then we denote the advantage of adversary $\mathcal{A}$ in $\mathcal{G}_0$ and $\mathcal{G}_1$ as $\mathbf{Adv}_{\mathcal{G}_0}$ and $\mathbf{Adv}_{\mathcal{G}_1}$, respectively. Finally,

suppose that there exists a PPT adversary $\mathcal{A}$ winning either of the games with non-negligible probability.

In game $\mathcal{G}_1$, since both z_1 and z_2 are randomly chosen from $\{0,1\}^n$, the advantage for the adversary to distinguish (z_1, y_1, z_2) from R_3 is equivalent to the advantage of distinguishing $(r^*, \mathsf{G}(r^*))$ from (r_1, r_2) where r^*, r_1, r_2 are all uniformly random. Hence, by winning game $\mathcal{G}_1$, $\mathcal{A}$ distinguishes $(r^*, \mathsf{G}(r^*))$ from (r_1, r_2) with non-negligible probability. This is an obvious contraction to the non-adaptive security of G.

Now consider the game $\mathcal{G}_0$. Assume that a PPT adversary $\mathcal{A}$ wins the game by distinguishing, with non-negligible probability,

$$Dist(0) : (z_1, y_1, z_2) = (\mathsf{F}(r), \mathsf{G}(\mathsf{F}(r)), \mathsf{F}(\mathcal{D}(\mathsf{G}(\mathsf{F}(r)))))$$

from a random triplet R_3 for $r \leftarrow_\$ \{0,1\}^n$. Then, consider the following distributions.

$$Dist(1) : (r_1, \mathsf{G}(r_1), \mathsf{F}(\mathcal{D}(\mathsf{G}(r_1))))$$

for $r_1 \leftarrow_\$ \{0,1\}^n$.

$$Dist(2) : (r_1, r_2, \mathsf{F}(\mathcal{D}(r_2)))$$

for r_1 and $r_2 \leftarrow_\$ \{0,1\}^n$.

$$Dist(3) : (r_1, r_2, r_3) = \mathsf{R}_3$$

for r_1, r_2 and $r_3 \leftarrow_\$ \{0,1\}^n$.

Since $\mathcal{A}$ distinguishes Dist(0) from Dist(3) with non-negligible probability, $\mathcal{A}$ distinguishes two of the intermediate distributions, Dist(i) and Dist(j), for $i \neq j$, with non-negligible probability. This is clearly a contradiction since the above distributions only negligibly deviate from each other by the the non-adaptive security of F and G.

The advantage of $\mathcal{A}$ to distinguish the transcript from R_3 is $\mathbf{Adv}_{\mathcal{G}_0}/2 + \mathbf{Adv}_{\mathcal{G}_1}/2$ which is negligible since both $\mathbf{Adv}_{\mathcal{G}_0}$ and $\mathbf{Adv}_{\mathcal{G}_1}$ are negligible. Thus, inequality (29) holds. □

Theorem 10 ([Pie06])**.** *Let* F *and* G *be* $(k-1)$*-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 for* $k \geq 2$.

4.2.2 Intuitions of Adaptively Insecure Sequential Composition of F and G under UTKA

In the following description of building our counter-example functions, we use the sequential version of γ-UTKA in which Bob's first message is *dependent* on Alice's first message. That is, Bob must wait for the first message α_1 from Alice in order to compute his first message β_1. See Protocol 6 for the overview of sequetial γ-UTKA.

Now, 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 $u_1, u_2, \cdots, u_{\gamma+2}, u^* \leftarrow_\$ \{0,1\}^n$. On Q_1, we define F and G as follows.

Alice	Transcript	Bob
$r_{\mathsf{A}} \leftarrow_{\$} \{0,1\}^n$		$r_{\mathsf{B}} \leftarrow_{\$} \{0,1\}^n$
$\alpha_1 \leftarrow \mathsf{A}_1(r_{\mathsf{A}})$	$\xrightarrow{\alpha_1}$	
	$\xleftarrow{\beta_1}$	$\beta_1 \leftarrow \mathsf{B}_1(r_{\mathsf{B}}, \alpha_1)$
$\alpha_2 \leftarrow \mathsf{A}_2(r_{\mathsf{A}}, \beta_1)$	$\xrightarrow{\alpha_2}$	
	$\xleftarrow{\beta_2}$	$\beta_2 \leftarrow \mathsf{B}_2(r_{\mathsf{B}}, \alpha_1, \alpha_2)$
	$\vdots$	
$\alpha_\gamma \leftarrow \mathsf{A}_\gamma(r_{\mathsf{A}}, \beta_1, \cdots, \beta_{\gamma-1})$	$\xrightarrow{\alpha_\gamma}$	
	$\xleftarrow{\beta_\gamma}$	$\beta_\gamma \leftarrow \mathsf{B}_\gamma(r_{\mathsf{B}}, \alpha_1, \cdots, \alpha_\gamma)$
secret key $sk \leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{A}}, \beta_1, \cdots, \beta_\gamma)$		secret key $sk \leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{B}}, \alpha_1, \cdots, \alpha_\gamma)$

Protocol 6: Sequential version of γ-UTKA

- 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^i_{F} based on Q_i as described above. F tests if $\mathsf{Dec}_{sk^i_{\mathsf{F}}}(\mathsf{Enc}_{sk^{i-1}_{\mathsf{G}}}(s_{(\gamma+1)(i-1)})) = s_{(\gamma+1)(i-1)}$. Obviously, $sk^i_{\mathsf{F}} \neq sk^{i-1}_{\mathsf{G}}$ with overwhelming probability since the keys are computed based on insufficient number of valid messages. Hence, F outputs $(\alpha_1, \cdots, \alpha_i, r_{(i+1)i}, \cdots, r_{(\gamma)i}, \mathsf{Enc}_{sk^i_{\mathsf{F}}}(r_{(\gamma+1)i}), r_{(\gamma+1)i}, r_{(\gamma+3)i})$. Similarly, G undertakes the

same course of computations: G computes messages and shared key, tests the equality and finally outputs $(\beta_1, \cdots, \beta_i,\ s_{(i+1)i}, \cdots, s_{(\gamma)i}, \mathsf{Enc}_{sk_{\mathsf{G}}^i}(s_{(\gamma+1)i}), s_{(\gamma+1)i}, s_{(\gamma+3)i})$. 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 $(\beta_1,\ \beta_2,\ \cdots,\ \beta_{\gamma-1},\ s_{\gamma(\gamma-1)},\ \mathsf{Enc}_{sk_{\mathsf{G}}^{\gamma-1}}(s_{(\gamma+1)(\gamma-1)}),\ s_{(\gamma+1)(\gamma-1)},\ u^*)$. 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 $(\alpha_1, \cdots, \alpha_\gamma, \mathsf{Enc}_{sk_{\mathsf{F}}^\gamma}(r_{(\gamma+1)\gamma}), r_{(\gamma+1)\gamma}, r_{(\gamma+3)\gamma})$ 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:

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

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:

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

4.2.3 Formal Construction of Non-Adaptively Secure Function F

We use three underlying primitives for the construction of F: PRP $\pi : K \times \{0,1\}^n \to \{0,1\}^n$, PRP $\tilde{\pi} : K \times (\{0,1\}^n)^3 \to (\{0,1\}^n)^3$ where key space K is $\{0,1\}^n$ and γ-UTKA $\Phi_u = (\mathsf{A}, \mathsf{B})$ described in Section 4.2.2. We define function $\mathsf{F} : (\{0,1\}^n)^{\gamma+3} \to (\{0,1\}^n)^{\gamma+3}$ to emulate the A of Φ_u via a black-box access to A. In addition, we define F to internally retain a private key k_{F} of length n. The formal construction is presented in Algorithm 6.

Construction of F

1. For any $I = (u_1, u_2, \ldots, u_{\gamma+3})$, then
   Output $(v_1, v_2, \ldots, v_{\gamma+3})$ where
   $r_{\mathsf{F}} \leftarrow \pi_{k_{\mathsf{F}}}(u_{\gamma+3})$
   $\alpha_1 \leftarrow \mathsf{A}_1(r_{\mathsf{F}})$
   FOR i from 2 to γ DO
   $\alpha_i \leftarrow \mathsf{A}_i(r_{\mathsf{F}}, u_1, u_2, \cdots, u_{i-1})$
   ENDFOR
   $sk \leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{F}}, u_1, u_2, \cdots, u_\gamma)$
   $(a_1, a_2, a_3) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(u_{\gamma+1}, u_{\gamma+2}, u_{\gamma+3})$

   (a) If $u_{\gamma+2} = \pi_{sk}^{-1}(u_{\gamma+1})$
      i. then $(v_1, v_2, \ldots, v_{\gamma+3}) \leftarrow (\alpha_1, \alpha_2, \cdots, \alpha_\gamma, \pi_{sk}(a_1), a_1, k_{\mathsf{F}})$
      ii. else $(v_1, v_2, \ldots, v_{\gamma+3}) \leftarrow (\alpha_1, \alpha_2, \cdots, \alpha_\gamma, a_1, a_2, a_3)$

Algorithm 6: The algorithm of function F

Claim 4.7. *The function* F *is secure against any non-adaptive PPT adversary* $\mathcal{A}(q,t)$*, running in time* t *and making at most* q *non-adaptive queries, where* t *and* q *are any polynomials of security parameter* n*.*

Proof. We prove this claim by showing that

$$\mathbf{Adv}^{\mathsf{F}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\mathsf{A}}_{\mathcal{A}}(q,t') + \frac{q}{2^n} \tag{30}$$

where $t' = t + poly(q)$, which accounts for the extra time costs resulting from our reduction.

Let $\mathcal{A}$ be a PPT adversary making non-adaptive q queries to function F. Since F on the same input query outputs the same output vector, assume that $\mathcal{A}$ does not input the same query twice. Let Q_1, Q_2, $\cdots$, Q_q denote the q non-adaptive queries that $\mathcal{A}$ generates.

Consider the event that a non-adaptive query evokes case 1(a)i of Algorithm 6. That is, for some $i \in [1,q]$, $Q_i = (u_1, u_2, \cdots, u_{\gamma+3})$ satisfies the condition of 'if statement' at the beginning of 1(a): $u_{\gamma+2} = \pi_{sk}^{-1}(u_{\gamma+1})$. sk is computed by A on u_1, u_2, $\cdots$, u_α and an initial random string denoted by r_{F} in Algorithm 6. Since F picks a pseudo-random string generated from $\pi_{k_{\mathsf{F}}}(u_{\gamma+3})$ for the initial random string, $\mathcal{A}$ does not know what r_{F} is as well as what the shared key sk is. Therefore, in order to evoke case 1(a)i (so the corresponding output reveals the secret key of F), $\mathcal{A}$ merely guesses $u_{\gamma+1}$ and $u_{\gamma+2}$ such that $u_{\gamma+2} = \pi_{sk}^{-1}(u_{\gamma+1})$. Since π is a permutation, there uniquely exists $u_{\gamma+1}$ for given $u_{\gamma+2}$ satisfying the condition. Therefore, the probability of each Q_i succeeding to evoke case 1(a)i is $1/2^n$. Consequently, the success probability would be $q/2^n$ with asking q non-adaptive queries, which is negligible in security parameter n. This case analysis accounts for the final term of inequality (30).

Consider the case that all q non-adaptive queries evoke case 1(a)ii, which occurs with overwhelming probability. Let the outputs generated by F on $Q_1, Q_2, \cdots, Q_{\gamma+1}$ be denoted by

$$(\alpha_{11}, \cdots, \alpha_{\gamma 1}, a_{11}, a_{21}, a_{31}), \cdots, (\alpha_{1q}, \cdots, \alpha_{\gamma q}, a_{1q}, a_{2q}, a_{3q}). \tag{31}$$

Towards a contradiction, assume that a PPT adversary $\mathcal{A}$ distinguishes distribution (31) from uniform random with at least non-negligible probability δ. That is, $\mathcal{A}$ distinguishes (31) from

$$(r_{11}, r_{21}, \cdots, r_{(\gamma+3)1}), \cdots, (r_{1q}, r_{2q}, \cdots, r_{(\gamma+3)q}) \tag{32}$$

where r_{ij}'s are uniformly chosen over $\{0,1\}^n$ for $1 \leq i \leq \gamma + 3$ and $1 \leq j \leq q$.

Define the i hybrid distribution denoted by H_i as

$$H_i = h_1, h_2, \cdots h_i, h_{i+1}, \cdots, h_q$$

where h_j is from distribution (31) if $j \leq i$ and h_k is from uniform distribution (32) if $k > i$.

Since $|H_q - H_0| \geq \delta$, there must exist $l \in [0, q]$ such that $|H_l - H_{l+1}| \geq \delta/q$. By using $\mathcal{A}$ as a subroutine, we can construct another PPT distinguisher $\mathcal{D}$ which distinguishes either the outputs of A or the outputs of $\tilde{\pi}$ from uniform randoms of the respectively same length with non-negligible probability as follows. Let x be a given distribution in $(\{0,1\}^n)^{\gamma+3}$, which we do not know whether it comes from (31) or (32). Now, $\mathcal{D}$ composes a hybrid distribution as

$$H_l(x) = h_1, h_2, \cdots h_l, x, h_{l+2}, \cdots, h_q$$

where h_j is from distribution (31) for $j \leq l - 1$ and h_k is from uniform distribution (32) for $k \geq l + 1$. Then, $\mathcal{D}$ feeds $\mathcal{A}$ with $H_l(x)$. If $\mathcal{A}$ tells $\mathcal{D}$ that $H_l(x)$ comes from H_l, then $\mathcal{D}$ outputs 0 which means that x is uniform random. If $\mathcal{A}$ tells $\mathcal{D}$ that $H_l(x)$ comes from H_{l+1}, then $\mathcal{D}$ outputs 1 which means that x is generated by F. Since $\mathcal{A}$'s answer is correct with non-negligible probability δ/q, $\mathcal{D}$ distinguishes the output generated by F from uniform random of the same length with non-negligible probability δ/q as well. In fact, this implies that $\mathcal{D}$ distinguishes either the outputs of A or the outputs of $\tilde{\pi}$ from uniform random as follows. Whenever $\mathcal{D}$ wants to know whether a given unknown distribution $(x_1, \cdots, x_\gamma)$ in $(\{0,1\}^n)^\gamma$ is generated by A or uniform random, $\mathcal{D}$ defines x as $(x_1, \cdots, x_\gamma, x_{\gamma+1}, x_{\gamma+2}, x_{\gamma+3})$ where $x_{\gamma+1}, x_{\gamma+2}$ and $x_{\gamma+3}$ are uniformly chosen over $\{0,1\}^n$. Then, $\mathcal{D}$ follows the procedure described above. Also, whenever $\mathcal{D}$ wants to know whether a given unknown distribution $(y_{\gamma+1}, y_{\gamma+2}, y_{\gamma+3})$ in $(\{0,1\}^n)^3$ is generated by $\tilde{\pi}$ or uniform random, $\mathcal{D}$ defines x as $(y_1, \cdots, y_\gamma, y_{\gamma+1}, y_{\gamma+2}, y_{\gamma+3})$ where $y_1, \cdots, x_\gamma$ are uniformly chosen over $\{0,1\}^n$. Similarly, $\mathcal{D}$ follows the procedure described above. One of these two distinguishing attempts must work with non-negligible probability δ/q since $|H_l - H_{l+1}| \geq \delta/q$. In either case, we encounter contradictions to indistinguishability of underlying primitives A and $\tilde{\pi}$. Therefore, the distinguishing advantage of $\mathcal{A}$ on F is upper-bounded by the distinguishing advantage of $\mathcal{A}$ on both A and $\tilde{\pi}$, which accounts for the first two terms in the RHS of inequality (30). □

4.2.4 Formal Construction of Non-Adaptively Secure Function G

The function G defined over $(\{0,1\}^n)^{\gamma+3}$ contains one secret key k_{G} and runs B of UTKA Φ_u described in Section 4.2.2 as a subroutine to emulate interactive communication (as Bob) with F (as Alice) via its inputs and outputs. The underlying primitives for the construction of function G are identical to those for the construction of function F. The formal construction of function G is given in Algorithm 7 below.

Construction of G

1. For any $I = (u_1, u_2, \ldots, u_{\gamma+3})$, then
   Output $(v_1, v_2, \ldots, v_{\gamma+3})$ where
   $r_{\mathsf{G}} \leftarrow \pi_{k_{\mathsf{G}}}(u_1)$
   FOR i from 1 to γ DO
   $\beta_i \leftarrow \mathsf{B}_i(r_{\mathsf{G}}, u_1, u_2, \cdots, u_i)$
   ENDFOR
   $sk \leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{G}}, u_1, u_2, \cdots, u_\gamma)$
   $(b_1, b_2, b_3) \leftarrow \tilde{\pi}_{k_{\mathsf{G}}}(u_{\gamma+1}, u_{\gamma+2}, u_{\gamma+3})$

   (a) If $u_{\gamma+2} = \pi_{sk}^{-1}(u_{\gamma+1})$,
      i. then $(v_1, v_2, \ldots, v_{\gamma+3}) \leftarrow (k_{\mathsf{G}}, u_{\gamma+3}, 0^n, \cdots, 0^n)$
      ii. else $(v_1, v_2, \ldots, v_{\gamma+3}) \leftarrow (\beta_1, \beta_2, \cdots, \beta_\gamma, \pi_{sk}(b_1), b_1, b_3)$

Algorithm 7: The algorithm of function G

Claim 4.8. *The function* G *is secure against any non-adaptive PPT adversary* $\mathcal{A}(q,t)$*, running in time* t *and making at most* q *non-adaptive queries, where* t *and* q *are any polynomials of security parameter* n*.*

Proof. We reduce the security of function G to the indistinguishability of π, $\tilde{\pi}$, the security of γ-UTKA Φ_u, and the probability of guessing secret key k_G by showing the following inequality:

$$\mathbf{Adv}^{\mathsf{G}}_{\mathcal{A}}(q,t) \leq \mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t') + \mathbf{Adv}^{\mathsf{B}}_{\mathcal{A}}(q,t') + \frac{q}{2^n} \tag{33}$$

where $t' = t + poly(q)$, which accounts for the extra time costs resulting from our reduction. Suppose that a PPT adversary $\mathcal{A}$ generates non-adaptive q queries over $(\{0,1\}^n)^{\gamma+3}$. Then, consider the case that a non-adaptive query $Q_i = (u_1, u_2, \cdots, u_{\gamma+3})$ evokes case 1(a)i of Algorithm 7, which reveals the secret key of G. It is easy to see that A can make only a mere guess on $u_{\gamma+1}$ and $u_{\gamma+2}$ such that $u_{\gamma+2} = \pi_{sk}^{-1}(u_{\gamma+1})$, which succeeds with probability $1/2^n$. Hence, the total probability of succeeding such $u_{\gamma+1}$ and $u_{\gamma+2}$ by making q queries is negligible $q/2^n$ accounting for the final term of RHS of inequality (33).

Consider the case that the non-adaptive q queries evoke 1(a)ii of Algorithm 7. Using the identical hybrid argument as in the proof of non-adaptive security of F in Section 4.2.4, we can show that there exists a PPT distinguisher D that uses another PPT adversary as a subroutine and distinguishes the outputs of G from uniform random as one of the following two cases must be true. First, $\mathcal{D}$ distinguishes the first γ coordinates of an output from uniform random, where the first γ coordinates are generated by B. This leads to a contradiction to the indistinguishability of B which accounts for term $\mathbf{Adv}^{\mathsf{B}}_{\mathcal{A}}(q,t')$ of inequality (33).

Then, this implies that $\mathcal{D}$ distinguishes the last three coordinates of an output from uniform random, where the last three coordinates are generated as follows. G first computes three pseudo-random strings a_1, a_2 and a_3 by $\tilde{\pi}_{k_{\mathsf{G}}}(u_{\gamma+1}, u_{\gamma+2}, u_{\gamma+3})$ where $u_{\gamma+1}, u_{\gamma+2}$ and $u_{\gamma+3}$ are the final three coordinates of the input query and outputs $(\pi_{sk}(a_1),\ a_1,\ a_3)$ for some unknown key sk. Since $\mathcal{D}$ distinguishes $(\pi_{sk}(a_1),\ a_1,\ a_3)$ from uniform random, $\mathcal{D}$ distinguishes either a_1 and a_3 from uniform random or $\pi_{sk}(a_1)$ from uniform random. If the former is true, $\mathcal{D}$ can actually distinguish $\tilde{\pi}$ from a uniform function by ignoring the second coordinate of outputs from $\tilde{\pi}$. If the latter is true, this directly enables $\mathcal{D}$ to distinguish π from a uniform function. Therefore, the both cases lead us to contradictions, which account for $\mathbf{Adv}^{\tilde{\pi}}_{\mathcal{A}}(q,t')$ and $\mathbf{Adv}^{\pi}_{\mathcal{A}}(q,t')$ of inequality (33). □

4.2.5 Adaptive Insecurity of Sequential Composition of F and G

Claim 4.9. *The sequential composition* $\mathsf{G}(\mathsf{F}(\cdot))$ *is breakable by* $\gamma+1$ *adaptive queries.*

Proof. To prove the claim, we will present a particular sequence of $\gamma+1$ adaptive queries and describe how to adaptively build them. For the standard notation, let $Q_i = (u_{1i}, u_{2i}, \cdots, r_{(\gamma+2)i})$ be the ith adaptive input query. For the clarity of proof, we define 'valid' and 'invalid' messages and shared keys as follows. We define the the first 'valid' message generated from function F (as Alice by using A), denoted by α_1^{val} to be the output of $\mathsf{A}_1(r_\mathsf{F})$ where r_F is the initial random string picked by F as F executes PRP π_{k_F} on a coordinate of the input query. Similarly, we define the first 'valid' message generated from function G (as Bob by using B) denoted by β_i^{val} to be $\mathsf{B}_1(r_\mathsf{G}, \alpha_1^{val})$. Inductively for $2 \le i \le \gamma$, we define the ith valid message message of F (resp. G) to be the output of $\mathsf{A}_i(r_\mathsf{F}, \beta_1^{val}, \cdots, \beta_{i-1}^{val})$ (resp. $\mathsf{B}_i(r_\mathsf{G}, \alpha_1^{val}, \cdots, \alpha_i^{val})$). Finally, we define a valid shared key of F (resp. G) denoted by sk_F^{val} (resp. sk_G^{val}) to be $\mathsf{A}_{\gamma+1}(r_\mathsf{F}, \beta_1^{val}, \cdots, \beta_\gamma^{val})$ (resp. $\mathsf{B}_{\gamma+1}(r_\mathsf{G}, \beta_1^{val}, \cdots, \beta_\gamma^{val})$). We define the ith 'invalid' messages and 'invalid' shared key of F to be the output of $\mathsf{A}_i(r_\mathsf{F}, u_1, \cdots, u_{i-1})$ for $2 \le i \le \gamma+1$ where there exists i such that $u_i \neq \beta_i$. We define the ith 'invalid' messages and 'invalid' shared key of F, respectively denoted by α_i^{inval} and sk_F^{inval}, to be the output of $\mathsf{A}_i(r_\mathsf{F}, u_1, \cdots, u_{i-1})$ for $2 \le i \le \gamma+1$ where there exists i such that $u_i \neq \beta_i$. Similarly, we define the ith 'invalid' messages of G, denoted by β_i^{inval}, to be the output of $\mathsf{B}_i(r_\mathsf{G}, u_1, \cdots, u_i)$ for $1 \le i \le \gamma$ where there exists i such that $u_i \neq \alpha_i$. Finally, we define the 'invalid' shared key of G, denoted by sk_G^{inval}, to be the output of $\mathsf{B}_{\gamma+1}(r_\mathsf{G}, u_1, \cdots, u_\gamma)$ where there exists $1 \le i \le \gamma$ such that $u_i \neq \alpha_i$. Recall that all the messages generated by A and B are indistinguishable from uniform random over $\{0,1\}^n$ due to the uniform transcript property of UTKA $\Phi_u = (\mathsf{A}, \mathsf{B})$ regardless of the validity of the messages.

Our first adaptive query to $\mathsf{G}(\mathsf{F}(\cdot))$ can be any random vector in $(\{0,1\}^n)^{\gamma+3}$. Let us define the first query to be

$$Q_1 = (u_1, u_2, u_3, \cdots, u_{\gamma+2}, u^*). \tag{34}$$

On Q_1, F computes a random seed $r_\mathsf{F} \leftarrow \pi_{k_\mathsf{F}}(u^*)$ for the initiation of key agreement. Then, F first computes $\alpha_1^{val} \leftarrow \mathsf{A}_1(r_\mathsf{F})$ and then computes the following.

$$\begin{aligned}
\alpha_2^{inval} &\leftarrow \mathsf{A}_2(r_\mathsf{F}, u_1) \\
\alpha_3^{inval} &\leftarrow \mathsf{A}_3(r_\mathsf{F}, u_1, u_2) \\
&\vdots \\
\alpha_\gamma^{inval} &\leftarrow \mathsf{A}_\gamma(r_\mathsf{F}, u_1, u_2, \cdots, u_{\gamma-1}) \\
sk_\mathsf{F}^{inval} &\leftarrow \mathsf{A}_{\gamma+1}(r_\mathsf{F}, u_1, u_2, \cdots, u_\gamma)
\end{aligned}$$

The above computations produce only invalid messages and shared key since $u_1, u_2, \cdots, u_\gamma$ are randomly chosen. That is, $u_i = \beta_i^{inval}$ for all i with overwhelming probability. F checks if $\pi^{-1}_{sk_\mathsf{F}^{inval}}(u_{\gamma+1}) = u_{\gamma+2}$. The condition happens to be satisfied with only negligible probability. Hence, F computes $(r_{11}, r_{21}, r_{31}) \leftarrow \tilde{\pi}_{k_\mathsf{F}}(u_{\gamma+1}, u_{\gamma+2}, u^*)$ outputs

$$(\alpha_1^{val}, \alpha_2^{inval}, \cdots, \alpha_\gamma^{inval}, \pi_{sk_\mathsf{F}^{inval}}(r_{11}), r_{11}, r_{21}). \tag{35}$$

Taking (35) as an input query, G computes $r_{\mathsf{G}} \leftarrow \pi_{k_{\mathsf{G}}}(\alpha_1^{val})$. Then G computes the following.

$$\begin{aligned}
\beta_1^{val} &\leftarrow \mathsf{B}_1(r_{\mathsf{G}}, \alpha_1^{val}) \\
\beta_2^{inval} &\leftarrow \mathsf{B}_2(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{inval}) \\
&\vdots \\
\beta_\gamma^{inval} &\leftarrow \mathsf{B}_\gamma(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{inval}, \cdots, \alpha_\gamma^{inval}) \\
sk_{\mathsf{G}}^{inval} &\leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{inval}, \cdots, \alpha_\gamma^{inval})
\end{aligned}$$

Then G checks to see if $\pi^{-1}_{sk_{\mathsf{F}}^{inval}}(\pi_{sk_{\mathsf{F}}^{inval}}(r_{11})) = r_{11}$ where $\pi_{sk_{\mathsf{F}}^{inval}}(r_{11})$ and r_{11} from (35). It is easy to see that the equality holds with only negligible probability since $sk_{\mathsf{F}}^{inval} = sk_{\mathsf{G}}^{inval}$ with only negligible probability. G computes $(s_{11}, s_{21}, s_{31}) \leftarrow \tilde{\pi}_{k_{\mathsf{G}}}(\pi_{sk_{\mathsf{F}}^{inval}}(r_{11}), r_{11}, r_{21})$. Then, G finalizes the first round computation by outputting

$$(\beta_1^{val}, \beta_2^{inval}, \cdots, \beta_\gamma^{inval}, \pi_{sk_{\mathsf{G}}^{inval}}(s_{11}), s_{11}, s_{21}). \tag{36}$$

Inductively, for all $2 \le i \le \gamma$, the ith adaptive query is of the following form:

$$Q_i = \underbrace{(\beta_1^{val}, \cdots, \beta_{i-1}^{val}, \beta_i^{inval}, \cdots, \beta_\gamma^{inval}, \pi_{sk_{\mathsf{G}}^{inval}}(s_{1(i-1)}), s_{1(i-1)}}_{\textit{from the output of the } (i-1) \textit{ round}} \underbrace{u_*}_{\textit{from (34)}}). \tag{37}$$

On Q_i, F undertakes the course of computations identical to those on Q_1. In the same way, G performs the same computations as on $\mathsf{F}(Q_1)$ described above. Hence, the output of the sequential composition on Q_i is

$$(\beta_1^{val}, \cdots, \beta_i^{val}, \beta_{i+1}^{inval}, \cdots, \beta_\gamma^{inval}, \pi_{sk_{\mathsf{G}}^{inval}}(s_{1i}), s_{1i}, s_{2i}). \tag{38}$$

By (37), the γth adaptive query Q_γ is

$$Q_\gamma = \underbrace{(\beta_1^{val}, \cdots, \beta_{\gamma-1}^{val}, \beta_\gamma^{inval}, \pi_{sk_{\mathsf{G}}^{inval}}(s_{1(\gamma-1)}), s_{1(\gamma-1),}}_{\textit{from the output of the } (\gamma-1) \textit{ round}} \underbrace{u^*}_{\textit{from (34)}}). \tag{39}$$

On Q_γ, F obtains the initial randomness $r_{\mathsf{F}} \leftarrow \pi_{k_{\mathsf{F}}}(u^*)$. Then, F computes all the messages and shared key based on Q_γ as follows:

$$\begin{aligned}
\alpha_1^{val} &\leftarrow \mathsf{A}_1(r_{\mathsf{F}}) \\
\alpha_2^{val} &\leftarrow \mathsf{A}_2(r_{\mathsf{F}}, \beta_1^{val}) \\
&\vdots \\
\alpha_\gamma^{val} &\leftarrow \mathsf{A}_\gamma(r_{\mathsf{F}}, \beta_1^{val}, \beta_2^{val}, \cdots, \beta_{\gamma-1}^{val}) \\
sk_{\mathsf{F}}^{inval} &\leftarrow \mathsf{A}_{\gamma+1}(r_{\mathsf{F}}, \beta_1^{val}, \beta_2^{val}, \cdots, \beta_{\gamma-1}^{val}, \beta_\gamma^{inval}).
\end{aligned}$$

Again, F checks if $\pi^{-1}_{sk_{\mathsf{F}}^{inval}}(\pi_{sk_{\mathsf{G}}^{val}}(s_{1(\gamma-1)})) = s_{1(\gamma-1)}$. Since the equality is not true with overwhelming probability, F computes $(r_{1\gamma}, r_{2\gamma}, r_{3\gamma}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(\pi_{sk_{\mathsf{G}}^{inval}}(s_{1(\gamma-1)}), s_{1(\gamma-1)}, s_{2(\gamma-1)})$. Finally, F outputs the following:

$$(\alpha_1^{val}, \alpha_2^{val}, \cdots, \alpha_\gamma^{val}, \pi_{sk_{\mathsf{F}}^{inval}}(r_{1\gamma}), r_{1\gamma}, r_{2\gamma}). \tag{40}$$

On (40), G also obtains its initial randomness $r_{\mathsf{G}} \leftarrow \pi_{k_{\mathsf{G}}}(\alpha_1)$ for key agreement. Notice that (40) contains all the valid messages $\alpha_1^{val}, \alpha_2^{val}, \cdots, \alpha_\gamma^{val}$, G can compute all the valid messages and shared key as follows:

$$\begin{aligned}
\beta_1^{val} &\leftarrow \mathsf{B}_1(r_{\mathsf{G}}, \alpha_1^{val}) \\
\beta_2^{val} &\leftarrow \mathsf{B}_2(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{val}) \\
&\vdots \\
\beta_\gamma^{val} &\leftarrow \mathsf{B}_\gamma(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{val}, \cdots, \alpha_\gamma^{val}) \\
sk_{\mathsf{G}}^{val} &\leftarrow \mathsf{B}_{\gamma+1}(r_{\mathsf{G}}, \alpha_1^{val}, \alpha_2^{val}, \cdots, \alpha_\gamma^{val})
\end{aligned}$$

Again, G checks to see if $\pi^{-1}_{sk_{\mathsf{G}}^{val}}(\pi_{sk_{\mathsf{F}}^{inval}}(r_{1\gamma})) = r_{1\gamma}$ where $\pi_{sk_{\mathsf{F}}^{inval}}(r_{1\gamma})$ and $r_{1\gamma}$ from (40). The equality is satisfied with only non-negligible probability due to the invalidity of sk_{F}^{inval} even if sk_{G}^{val} is the correctly shared key of G. Therefore, G computes $(s_{1\gamma}, s_{2\gamma}, s_{3\gamma}) \leftarrow \tilde{\pi}_{k_{\mathsf{G}}}(\pi_{sk_{\mathsf{F}}^{inval}}(r_{1\gamma}), r_{1\gamma}, r_{2\gamma})$. Then, G outputs:

$$(\beta_1^{val}, \beta_2^{val}, \cdots, \beta_\gamma^{val}, \pi_{sk_{\mathsf{G}}^{val}}(s_{1\gamma}), s_{1\gamma}, s_{2\gamma}). \tag{41}$$

Given (41), we define the final $(\gamma+1)$th adaptive input query $Q_{\gamma+1}$ as

$$Q_{\gamma+1} = (\underbrace{\beta_1^{val}, \beta_2^{val}, \cdots, \beta_\gamma^{val}, \pi_{sk_{\mathsf{G}}^{val}}(s_{1\gamma}), s_{1\gamma},}_{from(41)} \underbrace{u_*}_{from(34)}).$$

On $Q_{\gamma+1}$, F retrieves r_{F} from $\pi_{k_{\mathsf{F}}}(u^*)$. Now, F obtains all the valid messages β_i^{val} for all $1 \le i \le \gamma$ from $Q_{\gamma+1}$ so that G can compute all the valid messages α_i^{val} for all $1 \le i \le \gamma$ and shared key sk_{F}^{val} by the computations described above. F if $\pi^{-1}_{sk_{\mathsf{F}}^{val}}(\pi_{sk_{\mathsf{G}}^{val}}(s_{1(\gamma-1)})) = s_{1(\gamma-1)}$. As the equality holds since $sk_{\mathsf{F}}^{val} = sk_{\mathsf{G}}^{val}$, F is convinced that $Q_{\gamma+1}$ is adaptively generated query. Computing $(r_{1(\gamma+1)}, r_{2(\gamma+1)}, r_{3(\gamma+1)}) \leftarrow \tilde{\pi}_{k_{\mathsf{F}}}(\pi_{sk_{\mathsf{G}}^{val}}(s_{1\gamma}), s_{1\gamma}, s_{2\gamma})$, F outputs a vector that contains its secret key k_{F} as:

$$(\alpha_1^{val}, \alpha_2^{val}, \cdots, \alpha_\gamma^{val}, \pi_{sk_{\mathsf{F}}^{val}}(r_{1(\gamma+1)}), r_{1(\gamma+1)}, k_{\mathsf{F}}). \tag{42}$$

On (42), G can compute sk_{G}^{val} as in the previous rounds. G checks to see if $\pi^{-1}_{sk_{\mathsf{G}}^{val}}(\pi_{sk_{\mathsf{F}}^{val}}(r_{1(\gamma+1)}))$ $= r_{1(\gamma+1)}$. As the equality obviously holds, G is also convinced that $Q_{\gamma+1}$ is adaptively generated. Therefore, G obtains k_{F}, the secret key of F from (42), then G outputs the following vector:

$$(k_{\mathsf{G}}, k_{\mathsf{F}}, 0^n, \cdots, 0^n),$$

which reveals both secret keys of G and F. □

By Claim 4.7, 4.8, and 4.9, we obtain the following impossibility of adaptively secure sequential composition under the existence of UTKA.

Theorem 11. *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.*

Thus, putting the above theorem together with Pietrzak's result [Pie06], we have the equivalence of the above impossibility and the existence of UTKA as formally state below.

Theorem 12. *The sequential composition of two pseudo-random functions does not imply adaptive security if and only if the uniform-transcript key agreement exists.*

As we have proved the equivalence in the contexts of both parallel and sequential compositions by Theorem 9 and Theorem 12 respectively, we immediately obtain the following theorem which is our main theorem.

Theorem 13. *The 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

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

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

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

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 14. *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 corollaries.

Corollary 15. *If a family of dense trapdoor permutations exists, then the parallel self-composition of a non-adaptively secure function does not imply adaptive security.*

Corollary 16. *If a UTKA exists, then the parallel self-composition of a non-adaptively secure function does not imply adaptive security.*

In addition, the above construction of $\mathcal{F}$ defined in $(*)$ can be applied to non-adaptively secure pseudo-random functions F and G such that their sequential general-composition is adaptively insecure. 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. That is, $\mathcal{F}$ is the same as $(*)$ and we measure the probability to be equally $1/2$ that the outer function $\mathcal{F}$ has the first bit of seed equal to 0 and the first bit of seed equal to 1. Thus, $\mathcal{F}(\mathcal{F}(\cdot))$ is also adaptively insecure. Consequently, we easily 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 and 4 relevant to sequential composition insecurity from DTP and γ-UTKA, we derive the following corollaries.

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

Corollary 19. *If a UTKA exists, then the sequential self-composition of a non-adaptively secure function does not imply adaptive security.*

References

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *Advances in Cryptology (CRYPTO 08), Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *In Selected Areas in Cryptography, volume 2595 of LNCS*, pages 62–75. Springer-Verlag, 2002.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology (EUROCRYPT 01), volume 2045 of LNCS*, pages 93–118. Springer, 2001.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of ACM (JACM)*, 33(4):792–807, 1986.

[GL89] Oded Goldreich and Leonid Levin. A hard-core predicate for all one-way functions. In *21th ACM Symposium on the Theory Of Computing (STOC 89)*, pages 25–32, 1989.

[Gol01] Oded Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.

[Hai04] Iftach Haitner. Implementing oblivious transfer using collection of dense trapdoor permutations. In *In Proceedings of the 1st Theory of Cryptography Conference (TCC 04)*, pages 394–409. Springer, 2004.

[Hol05] Thomas Holenstein. Key agreement from weak bit agreement. In *37th ACM Symposium on the Theory Of Computing (STOC 05)*, pages 664–673, 2005.

[Hol06] Thomas Holenstein. *Strengthening key agreement using hard-core sets*. PhD thesis, ETH Zurich, 2006.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. *Structure in Complexity Theory Conference, Annual*, 0:134, 1995.

[LR86] Michael Luby and Charles Rackoff. Pseudo-random permutation generators and cryptographic composition. In *18th ACM Symposium on the Theory Of Computing (STOC 86)*, pages 356–363, 1986.

[MP04] Ueli Maurer and Krzysztof Pietrzak. Composition of random systems: When two weak make one strong. In *Theory of Cryptography (TCC 04), Lecture Notes in Computer Science*, volume 2951, pages 410–427, 2004.

[Mye04] Steven Myers. Black-box composition does not imply adaptively security. In *Advances in Cryptology (EUROCRYPT 04), Lecture Notes in Computer Science*, volume 3027, pages 189–206. Springer, 2004.

[Pie05] Krzysztof Pietrzak. Composition does not imply adaptive security. In *Advances in Cryptology (CRYPTO 05), Lecture Notes in Computer Science*, volume 3621, pages 55–65. Springer, 2005.

[Pie06] Krzysztof Pietrzak. Composition implies adaptive security in minicrypt. In *Advances in Cryptology (EUROCRYPT 04), Lecture Notes in Computer Science*, volume 4004, pages 328–338. Springer, 2006.

[SP92] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science(SFCS 92)*, pages 427–436. IEEE Computer Society, 1992.

[Vau03] Serge Vaudenay. Decorrelation: A theory for block cipher security. *Journal of Cryptology*, 16, 2003.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de