SCIS 2007 The 2007 Symposium on Cryptography and Information Security
Sasebo, Japan, Jan. 23-26, 2007
The Institute of Electronics, Information and Communication Engineers

Universally Composable Identification

Ryo Nishimaki * Yoshifumi Manabe * † Tatsuaki Okamoto * †

Abstract— Universally Composable (UC) security provides very strong security guarantees. It has been shown that many useful two party protocols cannot be UC-secure with no setup assumption, but can be in the Common Reference Strings (CRS) model. This paper, for the first time, investigates the security of identification schemes in the UC framework. Identification schemes are two party protocols, so they cannot be UC-secure with no setup assumption. However, we show that conventionally secure identification schemes are equivalent to UC-secure identification schemes. Since identification schemes have key generation algorithms, this result does not contradict the impossibility results of UC. This paper first defines the UC-security of identification, i.e., we define the ideal functionality of identification, $\mathcal{F}_{\mathrm{ID}}$. We then show that UC-secure identification is equivalent to conventionally secure identification.

Keywords: universal composability, identification, setup assumptions

1 Introduction

1.1 Background

Since Canetti introduced universal composability (UC) as a framework for analyzing the security of cryptographic primitives/protocols [Can01], UC-security has been a standard security notion [Can01, Can04, CF01, CK02]. UC-secure protocols maintain their security even when they run concurrently with arbitrary other protocols.

Since UC represents stronger security requirements, many conventionally secure protocols fail to meet UC security requirements. In particular, we cannot design secure two party protocols in the UC framework with no setup assumption (i.e., plain model) [Can01, CF01, CKL03, CLOS02, DDM$^+$06], while there are conventionally secure two party protocols (e.g., commitment and zero-knowledge proofs) with no setup assumption.

We know, however, that we can design UC-secure two party protocols with a setup assumption. Assume that we can access trusted *Common Reference Strings* (i.e., the CRS model), any two party protocol can be UC-secure [CLOS02].

The CRS model is a strong setup assumption, so many researchers are investigating weaker setup assumptions or relaxed definitions of UC-security, and are trying to design UC-secure two party protocols with them [BCNP04, CDPW07, PS04].

This paper investigates the security of identification (ID) schemes in the UC framework. ID schemes are quite useful cryptographic protocols [FFS88, Oka92, Sho99] in large network systems like the Internet in which users are not acquainted with one another. In such systems, users wish to authenticate themselves to other users. This is achieved by ID schemes. The security notion of ID was defined by Feige, Fiat and Shamir [FFS88]. We call this security notion FFS-security in this paper. As described above, conventionally secure two party protocols cannot be UC-secure with no setup assumption [CKL03]. ID schemes are two party protocols, so they cannot be UC-secure with no setup assumption. It is important to realize UC-secure ID schemes with weak setup assumptions. The UC security of ID schemes has not been investigated.

Note that users need to generate public keys and secret keys in advance to authenticate themselves in ID schemes as in public key encryption (PKE), digital signature (SIG), key encapsulation mechanism (KEM), and identity-based encryption (IBE). Since these primitives have key generation algorithms, strictly speaking, the known result that UC-secure PKE (resp. SIG, KEM, IBE) is equivalent to conventionally secure PKE (resp. SIG, KEM, IBE) is not considered in the plain model [Can01, Can04, NMO06a, NMO06b], Thus, we cannot apply the impossibility result [CKL03] to ID. That is, we have the following problems:

1. What is the security definition of ID in the UC framework (i.e., how to define an ideal functionality of ID)?
2. Is UC-secure ID equivalent to FFS-secure ID?

1.2 Our results

This paper answers the above problems:

1. This paper defines the UC-security of ID, i.e., we define the ideal functionality of ID, $\mathcal{F}_{\mathrm{ID}}$.
2. We show that UC-secure ID is equivalent to conventionally secure (FFS-secure) ID.

* Department of Social Informatics, Graduate School of Informatics, Kyoto University, Yoshidahonmachi, Sakyo-ku, Kyoto-shi, Japan, nisimaki@ai.soc.i.kyoto-u.ac.jp
† NTT Laboratories, 1-1 Hikari-no-oka, Yokosuka, Kanagawa, 239-0847, Japan, manabe.yoshifumi@lab.ntt.co.jp, okamoto.tatsuaki@lab.ntt.co.jp

2 Preliminaries

2.1 Notations and conventions

We describe probabilistic algorithms and experiments using standard notations and conventions. For probabilistic algorithm A, $A(x_1, x_2, ...; r)$ denotes the random variable of A's output on inputs $x_1, x_2, ...$ and coins r. We let $y \xleftarrow{R} A(x_1, x_2, ...)$ denote that y is randomly selected from $A(x_1, x_2, ...; r)$ according to its distribution.

We say that function $f : \mathbb{N} \to \mathbb{R}$ is negligible in security parameter k if, for every constant $c \in \mathbb{N}$, there exists $k_c \in \mathbb{N}$ such that $f(k) < k^{-c}$ for any $k > k_c$. Hereafter, we often use $f < \epsilon(k)$ to mean that f is negligible in k. On the other hand, $f > \mu(k)$ means that f is non-negligible in k, i.e., function $f : \mathbb{N} \to \mathbb{R}$ is non-negligible in k, if there exists a constant $c \in \mathbb{N}$ such that for every $k_c \in \mathbb{N}$, there exists $k > k_c$ such that $f(k) > k^{-c}$. A distribution ensemble $X = \{X(k, z)\}_{k \in \mathbb{N}, z \in \{0,1\}^*}$ is an infinite set of probability distributions, where a distribution $X(k, z)$ is associated with each $k \in \mathbb{N}$ and $z \in \{0, 1\}^*$. The ensembles considered in this paper describe outputs of computations where parameter z represents input, and parameter k is taken to be the security parameter. Two *binary* distribution ensembles X and Y are statistically indistinguishable (written $X \approx Y$) if for any $c, d \in \mathbb{N}$ there exists $k_c \in \mathbb{N}$ such that for any $k > k_c$ and any $z \in \cup_{\kappa \leq k^d} \{0, 1\}^\kappa$ we have:

$$|\Pr[X(k, z) = 1] - \Pr[Y(k, z) = 1]| < k^{-c}.$$

Let $\mathcal{P}$ and $\mathcal{V}$ be a linked pair of interactive Turing machines (ITMs), and suppose that all possible interactions of $\mathcal{P}$ and $\mathcal{V}$ on each common input terminate in a finite number of steps. We denote by $\langle \mathcal{P}, \mathcal{V} \rangle(x)$ the random variable representing the (local) output of $\mathcal{V}$ when interacting with machine $\mathcal{P}$ on common input x, when the random input to each machine is uniformly and independently chosen. (Indeed, this definition is asymmetric, since it considers only $\mathcal{V}$'s output.)

2.2 Definition of secure ID schemes

ID scheme

An ID scheme consists of two stages:

Initialization: In this stage, each user (e.g., $\mathcal{P}$) generates a secret key (e.g., sk) and a public key (e.g., pk) by using probabilistic polynomial-time (PPT) algorithm `Gen` on input of the key size. A link between each user and its public key is established. Note that some schemes commonly share a part of the public key among all users as a system parameter.

Operation: In this stage, any user can demonstrate its identity to a verifier by performing some ID protocol related to its public key, where the input for the verifier is the public key. At the conclusion of this stage, the verifier either outputs "Accept" or "Reject".

ID schemes consist of r rounds. One "round" means that the prover and the verifier send a message to each other. In the first round, the first message is sent by the prover. In the last round, the last message is sent by the prover. One "process" means that the prover and the verifier execute r rounds and the verifier outputs the decision.

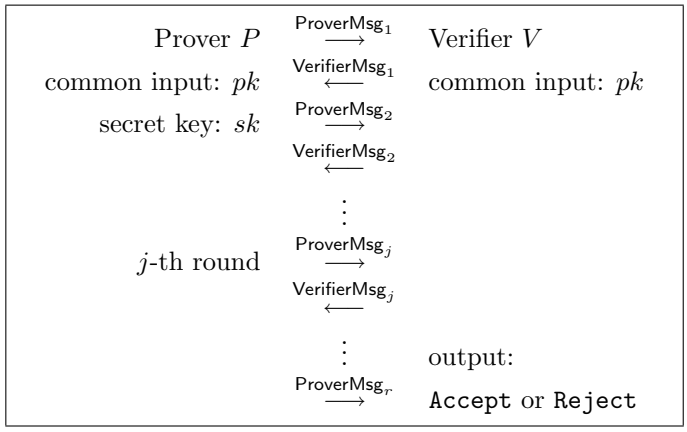

Figure 1: ID protocol

We review the definition of a secure ID scheme given by Feige, Fiat and Shamir [FFS88]. A prover (resp. verifier) is a honest prover denoted by $\mathcal{P}$ (resp. honest verifier denoted by $\mathcal{V}$), if it does not deviate from the protocols dictated by the scheme. Let $\mathcal{I}_P$ be a cheating prover who does not complete the Initialization stage as $\mathcal{P}$ and may deviate from the protocols. $\mathcal{I}_V$ is a cheating verifier.

FFS-secure ID. A FFS (Feige-Fiat-Shamir) secure ID scheme consists of a pair $(\mathtt{Gen}, \Pi)$, where `Gen` is a PPT algorithm and $\Pi = (\mathcal{P}, \mathcal{V})$ is a pair of PPT ITMs satisfying the following conditions:

Completeness: For any $k \in \mathbb{N}$ and any $(pk, sk) \xleftarrow{R} \mathtt{Gen}(1^k)$,

$$\Pr[\langle \mathcal{P}(sk), \mathcal{V} \rangle(pk) = \mathtt{Accept}] = 1$$

Soundness: Let $\mathcal{I} = (\mathcal{I}_V, \mathcal{I}_P)$ be an adversary; we say $\mathcal{I}$ is polynomial-time if both probabilistic algorithms $\mathcal{I}_V$ and $\mathcal{I}_P$ are polynomial-time. In the first stage, given the public key, the adversary interacts with honest prover $\mathcal{P}$ and tries to collect information to succeed in its impersonation with auxiliary input z. $\mathcal{I}_V$ can output the information, τ, which will be transferred to $\mathcal{I}_P$. In the second stage, the adversary interacts with honest verifier $\mathcal{V}$ with τ and auxiliary input z. We say adversary $\mathcal{I}$ breaks the FFS-security if she can make the honest verifier accept.

For any $k \in \mathbb{N}$ and any z let,

$$\mathbf{Adv}^{\text{id-aimp}}_{\Sigma,\mathcal{I}}(1^k) = \Pr[\mathbf{Exp}^{\text{id-aimp}}_{\Sigma,\mathcal{I}}(1^k) = 1]$$

$$\text{Experiment } \mathbf{Exp}^{\text{id-aimp}}_{\Sigma,\mathcal{I}}(1^k)$$

$$(pk, sk) \xleftarrow{R} \mathtt{Gen}(1^k);$$

$$\tau \xleftarrow{R} \mathcal{I}_V(z)^{\mathcal{P}(sk)}(pk);$$

$$d \xleftarrow{\mathsf{R}} \langle \mathcal{I}_P(z,\tau), \mathcal{V}\rangle(pk);$$
return 1 if $d = \texttt{Accept}$, 0 otherwise

$\mathcal{I}_V(z)^{\mathcal{P}(sk)}(pk)$ means that $\mathcal{I}_V$ interacts with $\mathcal{P}(sk)$ on common input pk (i.e., $\langle \mathcal{P}(sk), \mathcal{I}_V(z)\rangle(pk)$) for polynomially many times. We say that Σ is sound, if, for any $\mathcal{I}$, $\mathbf{Adv}^{\text{id-aimp}}_{\Sigma,\mathcal{I}}(1^k)$ is negligible.

2.3 Universal composability

The universally composable security framework allows the security properties of cryptographic tasks to be defined such that security is maintained under a general composition with an unbounded number of instances of arbitrary protocols running concurrently. Security in this framework is called universally composable (UC) security. Informally, we describe this framework as follows: (See [Can01] for more details.)

We consider the real life world, the ideal process world, and environment $\mathcal{Z}$ that tries to distinguish these two worlds.

The real life world.

In this world, there are adversary $\mathcal{A}$ and protocol π, which realizes a functionality among some parties. Let $\texttt{REAL}_{\pi,\mathcal{A},\mathcal{Z}}(k,z,\mathbf{r})$ denote the output of environment $\mathcal{Z}$ when interacting with adversary $\mathcal{A}$ and parties $P_1,...,P_n$ running protocol π (hereafter denoted as $(\mathcal{A},\pi)$) on security parameter k, auxiliary input z and random input $\mathbf{r} = (r_{\mathcal{Z}}, r_{\mathcal{A}}, r_1, ..., r_n)$ (z and $r_{\mathcal{Z}}$ for $\mathcal{Z}$, $r_{\mathcal{A}}$ for $\mathcal{A}$, r_i for party P_i). Let $\texttt{REAL}_{\pi,\mathcal{A},\mathcal{Z}}(k,z)$ denote the random variable describing $\texttt{REAL}_{\pi,\mathcal{A},\mathcal{Z}}(k,z,\mathbf{r})$ when $\mathbf{r}$ is uniformly chosen.

The ideal process world.

In this world, there are a simulator $\mathcal{S}$ that simulates the real life world, an ideal functionality $\mathcal{F}$, and dummy parties. Let $\texttt{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z,\mathbf{r})$ denote the output of environment $\mathcal{Z}$ when interacting with adversary $\mathcal{S}$ and ideal functionality $\mathcal{F}$ (hereafter denoted as $(\mathcal{S},\mathcal{F})$) on security parameter k, auxiliary input z and random input $\mathbf{r} = (r_{\mathcal{Z}}, r_{\mathcal{S}}, r_{\mathcal{F}})$ (z and $r_{\mathcal{Z}}$ for $\mathcal{Z}$, $r_{\mathcal{S}}$ for $\mathcal{S}$, $r_{\mathcal{F}}$ for party $\mathcal{F}$). Let $\texttt{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)$ denote the random variable describing $\texttt{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z,\mathbf{r})$ when $\mathbf{r}$ is uniformly chosen.

Let $\mathcal{F}$ be an ideal functionality and let π be a protocol. We say that π UC-realizes $\mathcal{F}$, if for any adversary $\mathcal{A}$, there exists simulator $\mathcal{S}$, such that for any environment $\mathcal{Z}$ we have:

$$\texttt{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \approx \texttt{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$$

where $\mathcal{A}$, $\mathcal{S}$ and $\mathcal{Z}$ are PPT ITMs.

3 Universally composable ID

3.1 The ID functionality $\mathcal{F}_{\text{ID}}$

We present the ideal ID functionality, $\mathcal{F}_{\text{ID}}$ in Figure 2. The idea of $\mathcal{F}_{\text{ID}}$ is as follows: If prover $\mathcal{P}$ is initialized and uncorrupted, it is accepted by honest verifier (completeness). If prover P' is not initialized, it is never accepted by honest verifier (soundness).

3.2 UC-secure ID is equivalent to FFS-secure ID

We describe how to translate an ID scheme $\Sigma = (\texttt{Gen}, \Pi)$ into protocol π_Σ in the present setting. This is done as follows:

Functionality $\mathcal{F}_{\text{ID}}$

Initialization Upon receiving $(\texttt{Initialize}, sid, \mathcal{P})$ from party $\mathcal{P}$, verify that $sid = (\mathcal{P}, sid')$ for some sid'. If not, then ignore the request. Else hand $(\texttt{Initialize}, sid, \mathcal{P})$ to the adversary. Upon receiving $(\texttt{PublicKey}, sid, \mathsf{pk})$ from the adversary, output $(\texttt{PublicKey}, sid, \mathsf{pk})$ to $\mathcal{P}$ and record $(\mathcal{P}, \mathsf{pk})$ in ID-Reg.

Operation

1. Upon receiving $(\texttt{Identify}, sid, P', V, \mathsf{pk}')$ from party P', if $P' = \mathcal{P}$ and $\mathsf{pk}' \neq \mathsf{pk}$ then output an error message. Else, send $(\texttt{Identify}, sid, P', V, \mathsf{pk}')$ to the adversary. Upon receiving $\texttt{ok}$ from the adversary, send $(\texttt{Identify}, sid, P', V, \mathsf{pk}')$ to V.
   (a) if $P' = \mathcal{P}$ and $\mathsf{pk}' = \mathsf{pk}$, record $(\mathcal{P}, V, \mathsf{true})$ in PV-List.
   (b) if $P' \neq \mathcal{P}$ and $\mathsf{pk}' = \mathsf{pk}$, record (P', V, false) in PV-List.
   (c) Else, record nothing.
2. Upon receiving $(\texttt{Verify}, sid, V, P')$ from party V, send $(\texttt{Verify}, sid, V, P')$ to the adversary and record $(V, P', \mathsf{verifying})$ in PV-List.
3. Upon receiving $(\texttt{Output}, sid, P', V, \mathsf{decision})$ from the adversary, $(\mathsf{decision} \in \{\texttt{Accept}, \texttt{Reject}\})$
   (a) if (P', V, false) and $(V, P', \mathsf{verifying})$ are recorded and $\mathcal{P}$ is uncorrupted, send $(\texttt{Reject}, sid)$ to V and erase (P', V, false) and $(V, P', \mathsf{verifying})$.
   (b) if (P', V, true) and $(V, P', \mathsf{verifying})$ are recorded, $\mathsf{decision} = \texttt{Accept}$ and $\mathcal{P}$ is uncorrupted (in this case $P' = \mathcal{P}$), send $(\texttt{Accept}, sid)$ to V and erase (P', V, true) and $(V, P', \mathsf{verifying})$.
   (c) Else, send $(\texttt{decision}, sid)$ to V and erase $(P', V, *)$ and $(V, P', *)$ (if there are).

Figure 2: The ideal ID functionality, $\mathcal{F}_{\text{ID}}$

protocol π_Σ

Initialize When party $\mathcal{P}$ receives $(\texttt{Initialize}, sid, \mathcal{P})$, it verifies that $sid = (\mathcal{P}, sid')$. If not, it ignores the input. Else, it runs algorithm $(sk, pk) \xleftarrow{\mathsf{R}} \texttt{Gen}(1^k)$, keeps the secret key sk, lets the key $\mathsf{pk} = pk$ and outputs $(\texttt{PublicKey}, sid, \mathsf{pk})$.

Operation

1. When party $\mathcal{P}$ receives $(\texttt{Identify}, sid, \mathcal{P}, \mathcal{V}, \mathsf{pk}')$, where $sid = (\mathcal{P}, sid')$, if $\mathsf{pk}' \neq \mathsf{pk}$ then outputs an error message. Else, it computes prover's first message $\mathsf{ProverMsg}_1$ according to the description

of ID schemes with common input pk and sends $\mathsf{ProverMsg}_1$ to party $\mathcal{V}$ (through adversary $\mathcal{A}$).

2. When party $\mathcal{V}$ receives $(\mathtt{Verify}, sid, \mathcal{V}, \mathcal{P})$, it computes verifier's first message $\mathsf{VerifierMsg}_1$ according to the description of ID schemes and $\mathsf{ProverMsg}_1$ (including pk) and sends $\mathsf{VerifierMsg}_1$ to party $\mathcal{P}$ if it has already received $\mathsf{ProverMsg}_1$.

3. When both $\mathcal{P}$ and $\mathcal{V}$ receive the first message from the other, they go on to interact with each other to complete the ID scheme.

4. When party $\mathcal{V}$ receives prover's last message, it decides whether to accept or reject according to the interaction. If the decision is $\mathtt{Accept}$ then $\mathcal{V}$ outputs $(\mathtt{Accept}, sid)$. Else, if the decision is $\mathtt{Reject}$ it outputs $(\mathtt{Reject}, sid)$.

Corrupt: When a party is corrupted, it reveals its internal state, which includes all past requests and answers, and for $\mathcal{P}$ also the state of proving algorithm (including the secret key and the randomness).

Simulator $\mathcal{S}$ Next, we present the activities of simulator $\mathcal{S}$. $\mathcal{S}$ invokes a simulated copy of $\mathcal{A}$ and proceeds as follows:

1. When $\mathcal{S}$ receives message $(\mathtt{Initialize}, sid, \mathcal{P})$ from $\mathcal{F}_{\mathrm{ID}}$, it runs $(pk, sk) \stackrel{\mathsf{R}}{\leftarrow} \mathtt{Gen}(1^n)$, lets $\mathsf{pk} = pk$, and returns $(\mathtt{PublicKey}, sid, \mathsf{pk})$.

2. When $\mathcal{S}$ receives message $(\mathtt{Identify}, sid, P', \mathcal{V}, \mathsf{pk}')$ from $\mathcal{F}_{\mathrm{ID}}$, if $P' = \mathcal{P}$ (in this case, $\mathsf{pk}' = \mathsf{pk}$), it simulates honest prover's first message by using secret key sk and given pk. Else, it simulates cheating prover's case. See 5. below. If the simulated copy of $\mathcal{A}$ discards the first message from the prover to the verifier, $\mathcal{S}$ does not send $\mathtt{ok}$ to $\mathcal{F}_{\mathrm{ID}}$. Else, it sends $\mathtt{ok}$.

3. When $\mathcal{S}$ receives message $(\mathtt{Verify}, sid, \mathcal{V}, P')$, it simulates verifier's first message by using pk'. If the verifier is corrupted, it simulates cheating verifier's case. See 6. below.

4. If $\mathcal{S}$ receives $(\mathtt{Identify}, sid, \mathcal{P}, \mathcal{V}, \mathsf{pk}')$ and $(\mathtt{Verify}, sid, \mathcal{P}, \mathcal{V})$, it simulates subsequent interactions between the prover and the verifier with common input pk. That is, it simulates $\langle \mathcal{P}(sk), \mathcal{V} \rangle(\mathsf{pk})$. If $\mathcal{A}$ does not discard or rewrite any messages, then $\mathcal{S}$ returns $(\mathtt{Output}, sid, \mathcal{P}, \mathcal{V}, \mathtt{Accept})$ without fail. Else, returns $(\mathtt{Output}, sid, \mathcal{P}, \mathcal{V}, \mathtt{decision})$ according to the interaction.

5. If the prover is corrupted and the verifier is honest, $\mathcal{S}$ sends $\mathtt{Identify}$ command to $\mathcal{F}_{\mathrm{ID}}$ in the name of the corrupted prover. $\mathcal{S}$ plays the role of the honest verifier. That is, $\mathcal{S}$ simulates $\langle \mathcal{I}_P(z, \tau), \mathcal{V} \rangle(\mathsf{pk}')$ where $\mathcal{I}_P$ is corrupted prover (i.e., simulated copy of $\mathcal{A}$). The honest verifier does not have any secret information, so $\mathcal{S}$ can simulate the honest verifier perfectly. $\mathcal{S}$ returns $(\mathtt{Output}, sid, \mathcal{I}_P, \mathcal{V}, \mathtt{decision})$ according to the interaction.

6. If prover $\mathcal{P}$ is honest and the verifier is corrupted, $\mathcal{S}$ plays the role of the honest prover. That is, $\mathcal{S}$ simulates $\langle \mathcal{P}(sk), \mathcal{I}_V \rangle(\mathsf{pk})$ where $\mathcal{I}_V$ is corrupted verifier (i.e., simulated copy of $\mathcal{A}$). $\mathcal{S}$ returns $(\mathtt{Output}, sid, \mathcal{P}, \mathcal{I}_V, \mathtt{decision})$ according to $\mathcal{A}$'s message. $\mathcal{S}$ has the prover's secret key, so it can simulate the honest prover perfectly.

7. When $\mathcal{A}$ corrupts some party P_j, $\mathcal{S}$ corrupts P_j in the ideal process. If P_j is prover, then $\mathcal{S}$ reveals the secret key sk (and potentially the internal state of $\mathcal{P}$, if such state exists) as the internal state of P_j.

8. If $\mathcal{A}$ discards some messages between $\mathcal{P}$ and $\mathcal{V}$, and they cannot go on the protocol, $\mathcal{S}$ does not send any message to $\mathcal{F}_{\mathrm{ID}}$.

Theorem. π_Σ UC-realizes $\mathcal{F}_{\mathrm{ID}}$ with respect to adaptive adversaries if and only if ID scheme Σ is FFS-secure.

Proof.

("only if" part):

We prove that if Σ is not FFS-secure, then π_Σ does not UC-realize $\mathcal{F}_{\mathrm{ID}}$. If Σ does not satisfy completeness, we can construct environment $\mathcal{Z}$ that can distinguish whether $(\mathcal{F}_{\mathrm{ID}}, \mathcal{S})$ or $(\pi_\Sigma, \mathcal{A})$. $\mathcal{Z}$ proceeds as follows:

1. Activates party $\mathcal{P}$ with $(\mathtt{Initialize}, sid, \mathcal{P})$ and obtains $(\mathtt{PublicKey}, sid, \mathsf{pk})$.

2. Activates party $\mathcal{P}$ with $(\mathtt{Identify}, sid, \mathcal{P}, \mathcal{V}, \mathsf{pk})$ and activates party $\mathcal{V}$ with $(\mathtt{Verify}, sid, \mathcal{V}, \mathcal{P})$.

3. Output 1 if receives $(\mathtt{decision}, sid)$ where $\mathtt{decision}$ is $\mathtt{Reject}$, otherwise outputs 0 and halts.

When $\mathcal{Z}$ interacts with $\mathcal{A}$ and π_Σ, $\mathcal{Z}$ receives $(\mathtt{decision}, sid)$ where $\mathtt{decision}$ is $\mathtt{Reject}$ with non-negligible probability, since initialized $\mathcal{P}$ is rejected with non-negligible probability by incompleteness. In contrast, when $\mathcal{Z}$ interacts with ideal process for $\mathcal{F}_{\mathrm{ID}}$ and any adversary, $\mathcal{Z}$ never receives $(\mathtt{decision}, sid)$ where $\mathtt{decision}$ is $\mathtt{Reject}$, since $\mathcal{P}$ is initialized prover. It follows that $\mathcal{Z}$ can distinguish with non-negligible probability.

If Σ does not satisfy soundness, in more detail, assuming that there exists adversary $\mathcal{I}^*$ that can impersonate $\mathcal{P}$ without $\mathcal{P}$'s secret information with non-negligible probability ϵ, we prove that there is real life adversary $\mathcal{A}$ such that for any ideal process adversary $\mathcal{S}$ there exists environment $\mathcal{Z}$ that can tell with non-negligible probability whether $(\pi_\Sigma, \mathcal{A})$ or $(\mathcal{F}_{\mathrm{ID}}, \mathcal{S})$ by using adversary $\mathcal{I}^*$. $\mathcal{Z}$ proceeds as follows:

1. $\mathcal{Z}$ corrupts two parties in advance. The corrupted parties are $\tilde{P}_{\mathcal{I}}$ and $\tilde{V}_{\mathcal{I}}$.

2. Activates party $\mathcal{P}$ with $(\mathtt{Initialize}, sid, \mathcal{P})$, obtain $(\mathtt{PublicKey}, sid, \mathsf{pk})$ and hands pk to $\mathcal{I}^*$.

3. Activates party $\mathcal{P}$ with $(\mathtt{Identify}, sid, \mathcal{P}, \tilde{V}_{\mathcal{I}}, \mathsf{pk})$, activates party $\tilde{V}_{\mathcal{I}}$ with $(\mathtt{Verify}, sid, \tilde{V}_{\mathcal{I}}, \mathcal{P})$ and plays the role of honest prover for adversary $\mathcal{I}^*$ (See below for more details).

4. Activates party $\tilde{P}_{\mathcal{I}}$ with $(\texttt{Identify}, sid, \tilde{P}_{\mathcal{I}}, \mathcal{V}, \mathsf{pk})$, activates party $\mathcal{V}$ with $(\texttt{Verify}, sid, \mathcal{V}, \tilde{P}_{\mathcal{I}})$ and plays the role of honest verifier for adversary $\mathcal{I}^*$ (See below for more details).

5. Outputs 1 if receives $(\texttt{decision}, sid)$ where decision is Accept, otherwise outputs 0 and halts.

In step 3, $\mathcal{Z}$ relays the message from $\mathcal{I}^*$ to corrupted party $\tilde{V}_{\mathcal{I}}$ through adversary $\mathcal{A}$ ($\mathcal{S}$). $\mathcal{A}$ makes $\tilde{V}_{\mathcal{I}}$ send the message to honest prover $\mathcal{P}$. When $\mathcal{A}$ receives the reply from $\mathcal{P}$, she passes it to $\mathcal{Z}$. $\mathcal{Z}$ relays $\mathcal{A}$'s message to $\mathcal{I}^*$. Thus, $\mathcal{I}^*$ can collect information from initialized $\mathcal{P}$, who has secret key sk and outputs τ.

In step 4, $\mathcal{Z}$ passes τ to $\mathcal{I}^*$ as an input and relays the message from $\mathcal{I}^*$ to corrupted party $\tilde{P}_{\mathcal{I}}$ through adversary $\mathcal{A}$ ($\mathcal{S}$). $\mathcal{A}$ makes $\tilde{P}_{\mathcal{I}}$ send the message to honest verifier $\mathcal{V}$. When $\mathcal{A}$ receives the reply from $\mathcal{V}$, she passes it to $\mathcal{Z}$. $\mathcal{Z}$ relays $\mathcal{A}$'s message to $\mathcal{I}^*$.

When $\mathcal{Z}$ interacts with $\mathcal{A}$ and π_Σ, $\mathcal{Z}$ receives $(\texttt{decision}, sid)$ where decision is Accept with non-negligible probability, since $\mathcal{I}^*$ can impersonate $\mathcal{P}$ without $\mathcal{P}$'s secret information with non-negligible probability (i.e., $\mathbf{Adv}^{\text{id-aimp}}_{\Sigma,\mathcal{I}^*} > \epsilon$.) and $\mathcal{Z}$ relayed $\mathcal{I}^*$'s messages. $\Pr[\mathcal{Z} \to 1 | \mathcal{Z} \leftrightarrow \texttt{REAL}]$ denotes the probability that $\mathcal{Z}$ outputs 1 when $\mathcal{Z}$ interacts with $\mathcal{A}$ and π_Σ.

$$\begin{aligned}\Pr[\mathcal{Z} \to 1 | \mathcal{Z} \leftrightarrow \texttt{REAL}] &= \Pr[\mathbf{Exp}^{\text{id-aimp}}_{\Sigma,\mathcal{I}^*}(1^k) = 1] \\ &> \epsilon\end{aligned}$$

In contrast, when $\mathcal{Z}$ interacts with ideal process for $\mathcal{F}_{\text{ID}}$ and any adversary, $\mathcal{Z}$ never receives $(\texttt{decision}, sid)$ where decision is Accept, since $\tilde{P}_{\mathcal{I}}$, who was activated with $(\texttt{Identify}, sid, \tilde{P}_{\mathcal{I}}, \mathcal{V}, \mathsf{pk})$, is not the initialized prover. $\Pr[\mathcal{Z} \to 1 | \mathcal{Z} \leftrightarrow \texttt{IDEAL}]$ denotes the probability that $\mathcal{Z}$ outputs 1 when $\mathcal{Z}$ interacts with $\mathcal{S}$ in the ideal process for $\mathcal{F}_{\text{IBE}}$.

$$\Pr[\mathcal{Z} \to 1 | [\mathcal{Z} \leftrightarrow \texttt{IDEAL}] = 0$$

Thus,

$$|\Pr[\mathcal{Z} \to 1 | \mathcal{Z} \leftrightarrow \texttt{REAL}] - \Pr[\mathcal{Z} \to 1 | \mathcal{Z} \leftrightarrow \texttt{IDEAL}]| > \epsilon$$

where ϵ is non-negligible. Therefore, $\mathcal{Z}$ can tell whether $(\mathcal{S}, \mathcal{F}_{\text{ID}})$ or $(\mathcal{A}, \pi_\Sigma)$ with non-negligible probability.

("if" part):

We show that if π_Σ does not securely realize $\mathcal{F}_{\text{ID}}$, then Σ is not FFS-secure. In more detail, we assume for contradiction that there is real life adversary $\mathcal{A}$ such that for any ideal process adversary $\mathcal{S}$ there exists environment $\mathcal{Z}$ that can tell whether $(\mathcal{F}_{\text{ID}}, \mathcal{S})$ or $(\pi_\Sigma, \mathcal{A})$ with non-negligible probability ϵ. We then show that there exists $\mathcal{I}^*$ whose advantage $\mathbf{Adv}^{\text{id-aimp}}_{\Sigma,\mathcal{I}^*}(k) > \epsilon/l$ in the experiment of soundness, where l is a polynomial. (We'll provide a precise definition of l later.)

First, we provide a high level overview. $\mathcal{S}$ provides perfect simulation except that the verification algorithm in the ideal world returns Reject if there is recorded entry $(\mathcal{I}_P, \mathcal{V}, \mathsf{false})$ for not initialized $\mathcal{I}_P$, while in the real world the verifier may accept $\mathcal{I}_P$ who is not initialized (i.e., $\langle \mathcal{I}_P(z,\tau), \mathcal{V} \rangle(\mathsf{pk}) = \texttt{Accept}$ may happen). If $\mathcal{Z}$ can detect the difference, this means that FFS-security is broken in the real world. Details are as follows:

We assume that Σ satisfies completeness (otherwise the theorem is proven). We show that it does not satisfy soundness, by constructing impersonator $\mathcal{I}^*$. $\mathcal{I}^*$ is given public key pk, runs a simulated copy of $\mathcal{Z}$, and simulates for $\mathcal{Z}$ an interaction with $\mathcal{S}$ in the ideal process for $\mathcal{F}_{\text{ID}}$. $\mathcal{I}^*$ runs a simulated copy of $\mathcal{A}$. There are n parties, $P_1, ..., P_n$.

1. When $\mathcal{Z}$ activates some party $\mathcal{P}$ with input $(\texttt{Initialize}, sid, \mathcal{P})$, $\mathcal{I}^*$ lets $\mathcal{P}$ output value pk calculated from pk.

2. When $\mathcal{Z}$ activates $\mathcal{P}$ with input $(\texttt{Identify}, sid, \mathcal{P}, \mathcal{I}_V, \mathsf{pk})$ and $\mathcal{I}_V$ with input $(\texttt{Verify}, sid, \mathcal{I}_V, \mathcal{P})$, $\mathcal{I}^*$ needs to simulate honest prover $\mathcal{P}$. $\mathcal{I}^*$ plays the role of cheating verifier (by using simulated copy of $\mathcal{A}$) and interacts with initialized prover $\mathcal{P}$ (i.e., $\langle \mathcal{P}(sk), \mathcal{I}^*(z) \rangle(pk)$), so it can simulate $\mathcal{P}$ by relaying messages from $\mathcal{P}$. Finally, it lets $\mathcal{I}_V$ return $(\texttt{decision}, sid)$ according to the decision of simulated copy of $\mathcal{A}$.

3. When $\mathcal{Z}$ activates $\mathcal{I}_P$ with input $(\texttt{Identify}, sid, \mathcal{I}_P, \mathcal{V}, \mathsf{pk})$ and $\mathcal{V}$ with input $(\texttt{Verify}, sid, \mathcal{V}, \mathcal{I}_P)$, $\mathcal{I}^*$ needs to simulate honest verifier $\mathcal{V}$. $\mathcal{I}^*$ plays the role of cheating prover (by using simulated copy of $\mathcal{A}$) and interacts with uncorrupted $\mathcal{V}$ (i.e., $\langle \mathcal{I}^*(\tau, z), \mathcal{V} \rangle(pk)$). Finally, it lets $\mathcal{V}$ return $(\texttt{decision}, sid)$ according to the decision of $\mathcal{V}$.

$\mathcal{Z}$ may activate multiple parties with Identify or Verify concurrently, so there are some differences between $\mathcal{Z}$'s attack scenario in UC framework (how to distinguish the real world from the ideal world) and $\mathcal{I}^*$'s attack scenario in FFS experiment (how to impersonate).

In UC, there are multiple parties and $\mathcal{Z}$ may run multiple processes concurrently. However, it suffices to consider only 4 parties, initialized prover $\mathcal{P}$, corrupted verifier $\mathcal{I}_V$, corrupted prover $\mathcal{I}_P$ and honest verifier $\mathcal{V}$, since $\mathcal{Z}$ cannot distinguish by using honest prover and honest verifier (from completeness of ID scheme) and all corrupted parties are under the control of the adversary (we can think that corrupted parties are identical to the adversary). Furthermore, it suffices to consider only two kinds of processes. One has initialized $\mathcal{P}$ interacting with corrupted $\mathcal{I}_V$ (we call this *TypeA* process). The other has corrupted $\mathcal{I}_P$ interacting with honest $\mathcal{V}$ (we call this *TypeB* process). The reason is as follows: $\mathcal{Z}$ cannot obtain information from the interaction between honest parties. That is, $\mathcal{Z}$ needs to activate both an honest party and a corrupted party.

$\mathcal{Z}$ runs multiple *TypeA* processes as a cheating verifier does in the FFS experiment of ID schemes. Furthermore, $\mathcal{Z}$ may run multiple *TypeB* processes. Note that, the cheating prover tries cheating only once in the FFS experiment of the ID schemes. However, the verifier does not have any secret information, so $\mathcal{I}^*$ can

simulate the verifier by herself without interacting with honest $\mathcal{V}$. Thus, $\mathcal{I}^*$ needs to interact with honest $\mathcal{V}$ to cheat $\mathcal{V}$ only when $\mathcal{Z}$ succeeds in distinguishing the two worlds. The probability that $\mathcal{I}^*$ can guess when $\mathcal{Z}$ succeeds is $1/l$ where l is the total number of *TypeB* process $\mathcal{Z}$ activates (polynomial).

Note that in step 2, $\mathcal{Z}$ cannot distinguish, since $\mathcal{S}$ offers perfect simulation (See case 6. of $\mathcal{S}$'s activities in Section 3.2).

Thus, $\mathcal{Z}$'s attack scenario is the same as $\mathcal{I}^*$'s attack scenario after all.

Let B denote the event that prover $\mathcal{P}$ generates public key pk (i.e., $\mathcal{P}$ is initialized), corrupted $\mathcal{I}_P$ is activated with input $(\texttt{Identify}, sid, \mathcal{I}_P, \mathcal{V}, \texttt{pk})$, uncorrupted $\mathcal{V}$ is activated with input $(\texttt{Verify}, sid, \mathcal{V}, \mathcal{I}_P)$, $\mathcal{P}$ is uncorrupted, and $\mathcal{Z}$ receives $(\texttt{decision}, sid)$ where $\texttt{decision} = \texttt{Accept}$. Corrupted $\mathcal{I}_P$ is never accepted in the ideal world, so $\mathcal{Z}$ cannot distinguish as long as event B does not occur (since $\mathcal{S}$ offers perfect simulation except for event B).

We assumed that $\mathcal{Z}$ can distinguish the real world from the ideal world with non-negligible probability ϵ, so event B occurs with non-negligible probability. The above observation ($\mathcal{Z}$'s attack scenario is the same as $\mathcal{I}^*$'s attack scenario), means that $\mathcal{I}^*$ succeeds in breaking FFS-security in the real world with non-negligible probability ϵ/l.

□

4 Conclusion

ID schemes are two party protocols that have key generation algorithms, so UC-secure ID is equivalent to conventionally secure ID. This is a rare case. Many results on UC provide us with insight into the limitation of UC. We know that large classes of functionalities cannot be UC-realized in the plain model [CKL03, Lin03, DDM+06]. To avoid this limitation, we need some setup assumptions [CLOS02, BCNP04, CDPW07] or non-standard assumptions [PS04, MMY06].

References

[BCNP04] B. Barak, R. Canetti, J. B. Nielsen, and R. Pass. Universally Composable Protocols with Relaxed Set-up Assumptions. *In Proc. of FOCS'04*, 2004.

[Can01] R. Canetti. Universally Composable Security: A New Paradigm for Cryptograpic Protocols. *In Proc. of FOCS'01*, 2001. Current Full Version Available at Cryptology ePrint Archive, Report 2000/067 http://eprint.iacr.org/.

[Can04] R. Canetti. Universally Composable Signatures, Certification, and Authenticated Communication. *In Proc. of 17th Computer Security Foundations Workshop*, 2004.

[CDPW07] R. Canetti, Y. Dodis, R. Pass, and S. Walfish. Universally Composable Security with Global Setup. *In Proc. of TCC'07 (To appear)*, 2007.

[CF01] R. Canetti and M. Fischlin. Universally Composable Commitments. *In Proc. of CRYPTO'01*, 2139 of LNCS, 2001.

[CK02] R. Canetti and H. Krawczyk. Universally Composable Key Exchange and Secure Channels. *In Proc. of EUROCRYPT'02*, 2332 of LNCS, 2002.

[CKL03] R. Canetti, E. Kushilevitz, and Y. Lindell. On the Limitations of Universally Composable Two-Party Computation Without Set-up Assumptions. *In Proc. of EUROCRYPT'03*, 2656 of LNCS, 2003.

[CLOS02] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally Composable Two-Party and Multi-Party Secure Computation. *In Proc. of STOC'02*, 2002.

[DDM+06] A. Datta, A. Derek, J. C. Mitchell, A. Ramanathan, and A. Scedrov. Games and the Impossibility of Realizable Ideal Functionality. *In Proc. of TCC'06*, 3876 of LNCS, 2006.

[FFS88] U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. *Journal of Cryptology*, 1:77-94, 1988.

[Lin03] Y. Lindell. General Composition and Universal Composability in Secure Multi-Party Computation. *In Proc. of FOCS'03*, 2003.

[MMY06] T. Malkin, R. Moriarty, and N. Yakovenko. Generalaized Environmental Security from Number Theoretic Assumptions. *In Proc. of TCC'06*, 3876 of LNCS, 2006.

[NMO06a] W. Nagao, Y. Manabe, and T. Okamoto. On the Equivalence of Several Security Notions of Key Encapsulation Mechanism. Cryptology ePrint Archive, Report 2006/268, 2006. http://eprint.iacr.org/.

[NMO06b] R. Nishimaki, Y. Manabe, and T. Okamoto. Universally Composable Identity-Based Encryption. *In Proc. of VIETCRYPT'06*, 4341 of LNCS, 2006.

[Oka92] T. Okamoto. Provable Secure and Practical Identification Schemes and Corresponding Signature Schemes. *In Proc. of CRYPTO'92*, 740 of LNCS, 1992.

[PS04] M. Prabhakaran and A. Sahai. New Notions of Security: Achieving Universal Composability without Trusted Setup. *In Proc. of STOC'04*, 2004.

[Sho99] V. Shoup. On the Security of a Practical Identification Scheme. *Journal of Cryptology*, 12:247-260, 1999.