On the Composability of Statistically Secure Bit Commitments

Rafael Dowsley* Jeroen van de Graaf† Jörn Müller-Quade‡
Anderson C. A. Nascimento§

November 4, 2008

Abstract

We show that stand-alone statistically secure commitments based on two-party stateless primitives are statistically universally composable. I.e. they are simulatable secure with an unlimited adversary, an unlimited simulator and an unlimited environment machine.

Especially, these protocols can be used in arbitrary statistically secure applications without lowering the security.

1 Introduction

Commitment Protocols: Commitment is one of the most fundamental cryptographic protocols. It is used as a sub-protocol in applications such as secure multi-party computation [17], contract signing [15] and zero-knowledge proofs [18, 16, 3]. A commitment protocol involves two players: the committer and the verifier. The idea behind the notion of commitment is simple: the committer provides the verifier with a digital equivalent of a "sealed envelope". This envelope should contain a value x in the commitment phase of the protocol. Before the committer helps the verifier in opening the envelope, the verifier should learn nothing about the value x. Additionally, the committer should not be able to change x after the commitment phase. When the committer helps the verifier in opening the envelope in the decommitment phase, the verifier learns the value x.

Universal Composability: A very large number of commitment protocols are known in the standalone setting, based on various assumptions, but this notion does not guarantee security when multiple copies of the protocol run at the same time, or when the commitment protocols are used within other protocols. Universally Composable (UC) Commitment [4, 5] is a notion of security that holds even when the commitment scheme is concurrently composed with an arbitrary set of protocols.

This notion of security is so strong that it is impossible to obtain UC commitment protocols if no set-up assumption is provided [5]. UC commitment protocols were constructed in the common

*Department of Electrical Engineering, University of Brasilia. Campus Universitario Darcy Ribeiro, Brasilia, CEP: 70910-900, Brazil. rafaeldowsley@redes.unb.br

†Departamento de Computação, Universidade Federal de Ouro Preto, Campus Universitário Morro do Cruzeiro, Ouro Preto/MG, CEP: 35400-000, Brazil. jvdg@iceb.ufop.br

‡Universitaet Karlsruhe, Institut fuer Algorithmen und Kognitive Systeme. Am Fasanengarten 5, 76128 Karlsruhe, Germany. muellerq@ira.uka.de

§Department of Electrical Engineering, University of Brasilia. Campus Universitario Darcy Ribeiro, Brasilia, CEP: 70910-900, Brazil. andclay@ene.unb.br

reference string (CRS) model [5, 6, 13, 11]. In the CRS model there exists an honestly generated random string at the system initialization; the simulator can generate its own string (as long as it looks indistinguishable from the honestly generated one). Barak et al. [2] show how to make the above schemes work in the PKI set-up model in the presence of a static adversary. In this model, parties have certified public keys. Dodis et al. [14] extend these results to adaptive corruptions. A UC commitment protocol was constructed in the random oracle model by Hofheinz and Müller-Quade [20]. Prabhakaran and Sahai [25] introduced a model in which the environment, the adversary and the simulator are given oracle access to super-polynomial angels. In this model one can securely implement any multiparty functionality without setup assumptions. In [22, 24] UC Commitment protocols based on tamper-proof hardware were proposed.

Statistically Secure Protocols and Our Result: There have been some questions on the equivalence of stand-alone and composable securities in the case of statistically secure protocols [23, 1]. In general, these securities notions are not equivalent [1]. Therefore, it is an interesting question to study if there are restricted scenarios where this equivalence holds. In this paper we show that commitment protocols based on stateless two-party primitives and matching a certain list of information-theoretical security properties commonly used in the literature (bindingness, concealingness and soundness) are not only secure in a simulation based way, but actually compose securely. As a particular consequence of our result, all the previously published commitment schemes based on noisy channels presented in the literature are actually secure when arbitrarily composed [7, 8, 12, 21].

Related Work: The question about the composability of statistically secure protocols has been previously addressed in [23, 1], where it was proven that the equivalence does not hold in general. In [9, 10] it was proven that perfectly secure oblivious transfer protocols according to a list of properties are *sequentially* composable, this result being extended to statistical security in [10]. The parallel composability of statistically-hiding commitments was shown in [19].

2 The UC Framework

This section briefly reviews the main concepts of the UC framework. We refer the reader to [4] for a more detailed explanation of this framework. We also describe the ideal functionalities that we use in this work.

2.1 Overview

In the UC framework, the security of a protocol to carry out a certain task is ensured in three phases:

1. One formalizes the framework, i.e., the process of executing a protocol in the presence of an adversary and an environment machine.
2. One formalizes an ideal protocol for carrying out the task in an ideal protocol using a "trusted party". In the ideal protocol the trusted party captures the requirements of the desired task and the parties do not communicate among themselves.
3. One proves that the real protocol emulates the ideal protocol. I.e. for every adversary in the real model there exists an ideal adversary in the ideal model such that no environment machine can distinguish if it is interacting with the real or with the ideal protocol.

The environment in the UC framework represents all activity external to the running protocol, so it provides inputs to the parties running the protocol and receives the outputs that the parties generate during the execution of the protocol. As stated above the environment also tries to distinguish between attacks on real executions of the protocol and simulated attacks against the ideal functionality. If no environment can distinguish the two situations, the real protocol emulates the ideal functionality.

Proving that a protocol is secure in the UC framework provides the following benefits:

1. The ideal functionality describes intuitively the desired properties of the protocol.
2. The protocols are secure under composition.
3. The security is retained when the protocol is used as a sub-protocol to replace an ideal functionality that it emulates.

The ideal protocol. An ideal functionality $\mathcal{F}$ represents the desired properties of a given task. Conceptually, $\mathcal{F}$ is treated as a local subroutine by the several parties that use it, and so the communication between the parties and $\mathcal{F}$ is supposedly secure (i.e., messages are sent by input and output tapes).

The ideal protocol for an ideal functionality $\mathcal{F}$ involves an ideal protocol adversary $\mathcal{S}$, an environment $\mathcal{Z}$ on input z and a set of dummy parties that interacts as defined below. Whenever a dummy party is activated with input x, it writes x onto the input tape of $\mathcal{F}$. Whenever the dummy party is activated with value x on its subroutine output tape, it writes x on subroutine output tape of $\mathcal{Z}$. The ideal protocol adversary $\mathcal{S}$ has no access to the contents of messages sent between dummy parties and $\mathcal{F}$, and it should send corruption messages directly to $\mathcal{F}$ that is responsible for determining the effects of corrupting any dummy party. The ideal functionality receives messages from the dummy parties by reading its input tape and sends messages to them by writing to their subroutine output tape. In the ideal protocol there is no communication among the parties using the adversary to deliver the message. The environment machine, $\mathcal{Z}$, can set the inputs to the parties, and read their outputs, but cannot see the communication with the ideal functionality.

The real protocol. In the real world, the protocol π is executed by parties $P_1, \ldots, P_n$ with some adversary $\mathcal{A}$ and an environment machine $\mathcal{Z}$ with input z. $\mathcal{Z}$ can set the inputs for the parties and see their outputs, but not the communication among the parties.

The parties of π can invoke subroutines, pass inputs to them and receive outputs from them. They can also write messages on the incoming communication tape of the adversary. These messages may specify the identity of the final destination of the message. $\mathcal{A}$ can send messages to any party ($\mathcal{A}$ delivers the message). In addition, they may use the ideal functionalities that are provided to the real protocol.

$\mathcal{A}$ can communicate with $\mathcal{Z}$ and the ideal functionalities that are provided to the real protocol. $\mathcal{A}$ can also corrupt parties. After receiving a special message (Corrupt *id*) from the environment, the adversary corrupt a party by delivering the message (Corrupt). By the definition of the process of corrupting, the environment always knows which parties are corrupted.

The adversarial model. The parties have unique identities and are locally PPT. The network is asynchronous without guaranteed delivery of messages. The communication is public, but authenticated (i.e., the adversary cannot modify the messages). The adversary is adaptive in corrupting parties, and is active in its control over corrupted parties. Any number of parties can be corrupted. Finally, the adversary, the environment and the simulator are allowed unbounded complexity. This assumption on the computational power of the simulator somehow weakens our result as the composition theorem cannot be applied several times if the real adversary were restricted to polynomial time, because the "is at least as secure as" relation cannot be proven to be transitive anymore. However, arbitrary composition is allowed when considering statistically secure protocols and this situation is common in the literature when proving general results on the composability of statistically secure protocols[1, 23, 10, 9].

Realizing an ideal functionality. We say that a protocol π statistically UC-realizes an ideal functionality $\mathcal{F}$ if for any real-life adversary $\mathcal{A}$ there exists an ideal-protocol adversary $\mathcal{S}$ such that no environment $\mathcal{Z}$, on any input z, can tell with non-negligible probability whether it is interacting with $\mathcal{A}$ and parties running π in the real-life process, or it is interacting with $\mathcal{S}$ and $\mathcal{F}$ in the ideal protocol. This means that, from the point of view of the environment, running protocol π is statistically indistinguishable of interacting with an ideal protocol for $\mathcal{F}$.

2.2 The Commitment Functionality

We present the ideal bit commitment functionality as described in [4] (a modified version of the first formalized functionality in [5]). The functionality is similar to the idea of a "sealed envelope" containing a value x. Before the committer helps the verifier in opening the envelope, the verifier learns nothing about the value x. But the committer cannot change the value after the commitment phase. When the committer helps the verifier in opening the envelope in the decommitment phase, the verifier learns the value x.

Functionality $\mathcal{F}_{COM}$

1. Upon receiving an input (COMMIT, *sid*, x) from $\mathcal{P}_1$, verify that $sid = (P_1, P_2, sid')$ for some $\mathcal{P}_2$, else ignore the input. Next, record x and generate a public delayed output (RECEIPT, *sid*) to $\mathcal{P}_2$. Once x is recorded, ignore any subsequent COMMIT inputs.

2. Upon receiving an input (OPEN, *sid*) from $\mathcal{P}_1$, proceed as follows: If there is a recorded value x then generate a public delayed output (OPEN, *sid*, x) to $\mathcal{P}_2$. Otherwise, do nothing.

3. Upon receiving a message (CORRUPT-COMMITTER, *sid*) from the adversary, send x to the adversary. Furthermore, if the adversary now provides a value x', and the RECEIPT output was not yet written on $\mathcal{P}_2$'s tape, then change the recorded value to x'.

The commitment phase is modeled in item 1 of the functionality in which $\mathcal{F}_{COM}$ receives the value committed to, records the value and send a public delayed output to the verifier to notify that a commitment was received (i.e. the message is first sent to the adversary, and later sent to the verifier when the confirmation from the adversary is received). The *sid* must contain the identities of the committer and verifier.

The decommitment phase takes place when the committer sends a message to $\mathcal{F}_{COM}$ to open the commitment as indicated an item 2 of $\mathcal{F}_{COM}$. If the committer has already recorded a value then a public delayed output with the value is generated to the verifier.

Item 3 of the functionality models the response when the adversary corrupts some committer. The $\mathcal{F}_{COM}$ sends the recorded value to the adversary and lets him modify the value if the RECEIPT message was not yet written to the verifier's tape.

2.3 Statistically Secure Two Party Stateless Functionality

We briefly introduce a hybrid model in which the participants have ideal access to a statistically secure two party stateless functionality $\mathcal{F}_{P_{V,W|X,Y}}$. Informally, this primitive receives $\mathcal{P}_1$'s input x taken from domain $\mathcal{X}$ and $\mathcal{P}_2$'s input y taken from domain $\mathcal{Y}$ and produces v and w defined over $\mathcal{V} \times \mathcal{W}$ according to the conditional probability distribution $P_{V,W|X,Y}$ and gives them to $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively.

Functionality $\mathcal{F}_{P_{V,W|X,Y}}$

1. Upon receiving an input (INPUT, *sid*, x) from $\mathcal{P}_1$, verify that $sid = (P_1, P_2, sid')$ for some $\mathcal{P}_1$ and $\mathcal{P}_2$, and that $x \in \mathcal{X}$, else ignore the input. Record *sid* and the corresponding x, if there is already a recorded value x for this *sid* ignore the input. If the values x and y corresponding to *sid* are already recorded, choose randomly v, w according to $P_{V,W|X,Y}$. Output v to $\mathcal{P}_1$ and w to $\mathcal{P}_2$.
2. Upon receiving an input (INPUT, *sid*, Y) from $\mathcal{P}_2$, verify that $sid = (P_1, P_2, sid')$ for some $\mathcal{P}_1$ and $\mathcal{P}_2$, and that $y \in \mathcal{Y}$, else ignore the input. Record *sid* and the corresponding y, if there is already a recorded value y for this *sid* ignore the input. If the values x and y corresponding to *sid* are already recorded, choose randomly v, w according to $P_{V,W|X,Y}$. Output v to $\mathcal{P}_1$ and w to $\mathcal{P}_2$.

In the ideal process for the functionality $\mathcal{F}_{COM}$ (described in section 2.2) $\mathcal{F}_{P_{V,W|X,Y}}$ is not used, so the ideal protocol adversary (simulator) that simulates a real-life adversary can play the role of $\mathcal{F}_{P_{V,W|X,Y}}$ for the simulated adversary.

3 Commitment Based on Statistically Secure Two Party Stateless Functionalities

In this section we define a stand-alone security model for commitment protocols based upon the previously defined two party stateless functionalities. A bit commitment protocol has two phases:

commitment and decommitment. In both phases, the committer and verifier, also denoted $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, have two channels available between them:

- a bidirectional authenticated noiseless channel, denoted as $\mathcal{F}_{AUTH}$, and
- the two party stateless functionality $\mathcal{F}_{P_{V,W|X,Y}}$.

We model the probabilistic choices of $\mathcal{P}_1$ by a random variable R_1 and those of $\mathcal{P}_2$ by a random variable R_2, so we can use deterministic functions in the protocol. Note that in this way, all the messages generated by $\mathcal{P}_1$ and $\mathcal{P}_2$ are well-defined random variables, depending on the bit value $\mathcal{P}_1$ wants to commit to, b. As usual, we assume that the noiseless messages exchanged by the players and their personal randomness are taken from $\{0,1\}^*$.

Commitment Phase: $\mathcal{P}_1$ has an input bit $b \in \{0,1\}$ that it wants to commit to. The protocol has some rounds of noiseless and authenticated communications between $\mathcal{P}_1$ and $\mathcal{P}_2$. After the i-th round, $\mathcal{P}_1$ and $\mathcal{P}_2$ input symbols x_i and y_i to the functionality $\mathcal{F}_{P_{V,W|X,Y}}$, which generates outputs v_i and w_i for $\mathcal{P}_1$ and for $\mathcal{P}_2$, according to $P_{V,W|X,Y}$. Let K denote all the noiseless messages exchanged between the players, and K^i the noiseless messages sent until round i. Denote similarly by x^i, y^i, v^i and w^i the vectors of these variables until round i. At the end of this commitment phase, $\mathcal{P}_2$ outputs (RECEIPT, *sid*). Note that this modeling allows multiple use of the $\mathcal{F}_{P_{V,W|X,Y}}$ functionality within a protocol in an interactive manner; let t denote the number of times the parties use this functionality.

Decommitment Phase: The parties exchange messages only over the noiseless channel. The committer, $\mathcal{P}_1$, announces the value b' that it claims that was the value it committed to in the first phase, his private randomness r_1, the outputs v^t generate by the functionality $\mathcal{F}_{P_{V,W|X,Y}}$ and his inputs x^t. Then $\mathcal{P}_2$ executes his test $\beta(x^t, y^t, v^t, w^t, r_2, r_1, k, b')$ for a deterministic function β. The test can accept or reject b'. If $\mathcal{P}_2$ accepts b', it outputs (OPEN, *sid*, b').

This modeling does not reflect all possible decommit protocols as one could e.g. use the functionality $\mathcal{F}_{P_{V,W|X,Y}}$ in the decommit phase. However such a decommit is always possible and we can prove our results even when restricting to this type of decommit protocols. Note that we could split the noiseless messages k in those originated by $\mathcal{P}_1$ and $\mathcal{P}_2$, but this will not be necessary in our proofs.

We call the view of $\mathcal{P}_2$ all the data in his possession after the completion of the commitment phase, i.e. y^t, w^t, r_2, k and denote it by $View_2$. $View_1$ is defined similarly.

We will now define the security of a protocol. A protocol is ϵ-*concealing* if for any possible behavior of $\mathcal{P}_2$ in the commitment phase,

$$\frac{1}{2} \cdot \sum_{r_1, r_2 \in \{0,1\}^*} \Pr[r_1] \cdot |\Pr[y^t, w^t, r_2, k | r_1, b=0] - \Pr[y^t, w^t, r_2, k | r_1, b=1]| \leq \epsilon$$

A protocol is δ-*sound-and-binding*, if for an honest verifier and committer executing the protocol and $b \in \{0,1\}$,

$$\Pr[\beta(X^t, Y^t, V^t, W^t, R_2, R_1, K, b) = accept] \geq 1 - \delta$$

and, for every possible private randomness used by $\mathcal{P}_1$ during the commit phase:

$$\Pr[\beta(X^t, Y^t, V^t, W^t, R_2, R_1, K, 0) = accept \ \& \ \beta(X'^t, Y^t, V'^t, W^t, R_2, R'_1, K, 1) = accept] \leq \delta$$

We call the protocol *stand-alone secure* if ϵ and δ are negligible in t.

4 Statistically Secure Universal Composability of Stand-Alone Secure Commitments

In this section we address the question of whether the stand-alone conditions for commitment protocols we stated previously are enough for ensuring their statistical universal composability.

We will now prove two lemmas that we will use later to prove the main result of this paper. We first show that, in any stand-alone secure bit commitment protocol based on two party stateless functionalities, given $\mathcal{P}_1$'s input to the functionality and all the noiseless communication exchanged by $\mathcal{P}_1$ and $\mathcal{P}_2$, one can almost surely infer $\mathcal{P}_1$'s commitment (this property is known as extractability).

Lemma 1. *Any stand-alone secure protocol of commitment based on two party stateless functionality supports extraction of committer's commitment given her input to the channel and all the noiseless communication.*

Proof. Every possible commit value b, and $\mathcal{P}_1$ and $\mathcal{P}_2$'s private randomness $R_1 = r_1$ and $R_2 = r_2$, induce a set of possible messages (also called conversation) to be sent by $\mathcal{P}_1$ that has a certain probability of being accepted, which we denote by ρ_{b,r_1,r_2}. By soundness we have

$$\sum_{r_1,r_2} Pr(r_1, r_2)\rho_{b,r_1,r_2} \geq 1 - \delta \tag{1}$$

Applying Chebyshev inequality we obtain that for each possible value of b there must exist a set of values of r_1 and r_2 which happens with total probability larger or equal than $1 - \delta$ for which $\rho_{b,r_1,r_2} \geq 1 - 2\sqrt{\delta}$. We call those sets *typical for* b and committer's conversations induced by r_1, r_2 in those sets *typical conversations for* b, here on denoted $\mathcal{C}_b$. Clearly, for each value of b, we have that $P(\mathcal{C}_b) \geq 1 - 2\sqrt{\delta}$.

From the bindingness condition we also obtain that the probability that a (cheating) $\mathcal{P}_1$ produces a conversation belonging to $\mathcal{C}_b$ should be negligible, when committing to a certain $b' \neq b$. Then we have an estimator for computing b from $\mathcal{P}_1$'s conversation. For each $b \in \{0, 1\}$ and for a certain δ, we compute $\mathcal{C}_b$, the set of typical committer conversations for b for which

$$\rho_{b,r_1,r_2} \geq 1 - 2\sqrt{\delta} \tag{2}$$

and check in which typical set the committer's conversation is contained. If none is found or if it is present in more than one set, then $\mathcal{P}_2$ declares an error. Clearly, the probability of error will be negligible if δ is. □

We now prove that, for any possible view of $\mathcal{P}_2$ after the commit phase is finished, there exists at least one view of $\mathcal{P}_1$ that passes the test executed by $\mathcal{P}_2$ at the end of the opening phase, for any possible committed value b (the so-called equivocability property of the commitment protocol).

Lemma 2. *Any stand-alone secure protocol of bit commitment based on two party stateless functionality has the equivocability property.*

Proof. After the commitment phase, $\mathcal{P}_2$ possesses y^t, w^t, r_2, k. Let the honest committer's opening information be x^t, v^t, r_1, b. From the correctness property of the protocol we know that

$$\Pr[\beta(x^t, y^t, v^t, w^t, r_2, r_1, k, b) = accept] \geq 1 - \delta$$

.

We claim that there exist $\tilde{x}^t, \tilde{v}^t, \tilde{r_1}$, so that

$$\Pr[\beta(\tilde{x}^t, y^t, \tilde{v}^t, w^t, r_2, \tilde{r_1}, k, b') = accept] \geq 1 - \delta,$$

because if this were not the case, $\mathcal{P}_2$ could break the concealing condition, computing

$$\Pr[\beta(x^t, y^t, v^t, w^t, r_2, r_1, k, b) = accept]$$

for all the possible values of x^t, v^t, r_1 and b and $\mathcal{P}_1$'s correct commitment would be the only one that would produce an overwhelmingly acceptance probability. □

We now use lemmas 1 and 2 to prove our main result, which is stated below:

Theorem 3. *Any stand-alone secure protocol of commitment based on two party stateless functionality statistically UC-realize $\mathcal{F}_{COM}$ using $\mathcal{F}_{AUTH}$ and $\mathcal{F}_{P_{V,W|X,Y}}$.*

Proof. We construct the ideal-protocol adversary $\mathcal{S}$ as follows. $\mathcal{S}$ runs a simulated copy of $\mathcal{A}$ in a black-box way, plays the role of the ideal functionality $\mathcal{F}_{P_{V,W|X,Y}}$ and simulates a copy of the hybrid interaction of π for the simulated adversary $\mathcal{A}$. In addition, $\mathcal{S}$ forwards all inputs from $\mathcal{Z}$ to $\mathcal{A}'s$ input and all outputs from $\mathcal{A}$ to $\mathcal{Z}$. $\mathcal{S}$ should be able to extract the committed value from the messages that it receives from $\mathcal{A}$ if $\mathcal{P}_1$ is corrupted and also should be able to send a commitment in the hybrid interaction and later open it to any value. These two properties are guaranteed by the lemmas 1 and 2. Below we describe the procedures of the simulator in each occasion:

Commitment - Uncorrupted Committer: If the environment $\mathcal{Z}$ writes a message (COMMIT, *sid*, *b*) on the input tape of an uncorrupted committer $\mathcal{P}_1$, then $\mathcal{P}_1$ copies the message to the functionality $\mathcal{F}_{COM}$, and $\mathcal{S}$ is informed about the commitment. $\mathcal{S}$ chooses a random bit b' to commit in the hybrid interaction and proceeds as follows:

- If $\mathcal{P}_2$ is uncorrupted, $\mathcal{S}$ simulates for $\mathcal{A}$ the messages that both parties send in each round of the noiseless communication (i.e., $\mathcal{S}$ chooses the randomness r_1 of $\mathcal{P}_1$ and the randomness r_2 of $\mathcal{P}_2$ and simulates the noiseless messages and $\mathcal{F}_{P_{V,W|X,Y}}$ in each round without revealing the inputs and outputs of the functionality to the adversary). When $\mathcal{A}$ delivers all noiseless messages, $\mathcal{S}$ allows $\mathcal{F}_{COM}$ to output (RECEIPT, *sid*) to $\mathcal{P}_2$ in the ideal protocol.
- If $\mathcal{P}_2$ is corrupted, $\mathcal{S}$ simulates for $\mathcal{A}$ the messages of an honest committer in each round of noiseless communication (i.e., $\mathcal{S}$ chooses the randomness r_1 of $\mathcal{P}_1$ and simulates the noiseless messages). After the adversary $\mathcal{A}$ delivers all the noiseless messages in the round i and sends the input $y_i \in Y$ of the corrupted verifier to $\mathcal{F}_{P_{V,W|X,Y}}$, $\mathcal{S}$ simulates the outputs of the functionality according to the probability $P_{V,W|X,Y}$ (using $x_i \in X$ that is determined by r_1 and b' as $\mathcal{P}_1$ input) and sends w_i to the adversary. The simulator proceeds to the next round.

Decommitment - Uncorrupted Committer: If $\mathcal{Z}$ writes a message (OPEN, *sid*) on the input tape of some uncorrupted committer $\mathcal{P}_1$, then $\mathcal{P}_1$ copies the message to the functionality $\mathcal{F}_{COM}$. If $\mathcal{P}_1$ has previously committed to a value b, $\mathcal{S}$ will receive the bit b. By lemma 2, $\mathcal{S}$ can find $\tilde{x}^t$, $\tilde{v}^t$ and $\tilde{r_1}$ such that after the exchange of messages in the decommitment phase, the bit b will be accepted in an honest verifier's test. $\mathcal{S}$ proceeds as follows:

- If $\mathcal{P}_2$ is uncorrupted, $\mathcal{S}$ simulates the messages of both parties in this phase and allows $\mathcal{F}_{COM}$ to output (OPEN, *sid*, b) to $\mathcal{P}_2$ in the ideal protocol when $\mathcal{A}$ delivers all messages.
- If $\mathcal{P}_2$ is corrupted, $\mathcal{S}$ simulates the messages sent by $\mathcal{P}_1$ in this phase.

Commitment - Corrupted Committer: If $\mathcal{A}$ lets some corrupted $\mathcal{P}_1$ commit to a bit b, $\mathcal{S}$ can extract b according to lemma 1. $\mathcal{S}$ proceeds as follows:

- If $\mathcal{P}_2$ is uncorrupted, $\mathcal{S}$ simulates in each round the noiseless messages sent by the honest verifier and $\mathcal{P}_2$'s input y_i to $\mathcal{F}_{P_{V,W|X,Y}}$ (i.e., $\mathcal{S}$ chooses the randomness r_2 of $\mathcal{P}_2$ and sends these messages). After $\mathcal{A}$ sends $\mathcal{P}_1$'s input x_i to the functionality, $\mathcal{S}$ simulates the outputs of $\mathcal{F}_{P_{V,W|X,Y}}$ and sends v_i to $\mathcal{A}$. When all the t rounds are finished and the simulated verifier accepts the commitment, then $\mathcal{S}$ sends the message (COMMIT, *sid*, b) to $\mathcal{F}_{COM}$.
- If $\mathcal{P}_2$ is corrupted, $\mathcal{S}$ just simulates $\mathcal{F}_{P_{V,W|X,Y}}$.

Decommitment - Corrupted Committer: If $\mathcal{A}$ tells some corrupted $\mathcal{P}_1$ to open a valid commitment with bit b' and $\mathcal{P}_2$ is uncorrupted, then $\mathcal{S}$ simulates the messages sent by the honest verifier in the decommitment interaction with $\mathcal{A}$. It then checks b' following the procedures used by an honest verifier in the hybrid interaction. If an honest verifier would reject it, then $\mathcal{S}$ stops; otherwise $\mathcal{S}$ sends (OPEN, *sid*) to $\mathcal{F}_{COM}$.

Corruption - Committer: If $\mathcal{A}$ corrupts $\mathcal{P}_1$, then $\mathcal{S}$ corrupts the committer in the ideal protocol and learns b. $\mathcal{S}$ proceeds as follows:

- If the (RECEIPT, *sid*) output was not written on $\mathcal{P}_2$'s tape before the corruption, then $\mathcal{A}$ has not yet delivered all the messages from the commitment phase and $\mathcal{A}$ can play the role of the committer in the remaining rounds of this phase. $\mathcal{S}$ uses the equivocability property of lemma 2 to find valid $\tilde{x}^i$, $\tilde{v}^i$ and $\tilde{r_1}$ for the i rounds already finished, follows in the remaining rounds the same procedures as in the case of commitment with corrupted committer, extracts the new value b' and sends b' to $\mathcal{F}_{COM}$.
- If the (RECEIPT, *sid*) output was written on $\mathcal{P}_2$'s tape before the corruption, the adversary knows k and possibly y^t, w^t and r_2 (only if $\mathcal{P}_2$ is already corrupted). $\mathcal{S}$ finds a valid $\tilde{x}^i$, $\tilde{v}^i$ and $\tilde{r_1}$ using the equivocability property of lemma 2 and sends them to $\mathcal{A}$.

Corruption - Verifier: If $\mathcal{A}$ corrupts $\mathcal{P}_2$, then $\mathcal{S}$ corrupts the verifier in the ideal protocol. $\mathcal{S}$ proceeds as follows:

- If $\mathcal{P}_2$ is corrupted after round i of the commitment stage and before the decommitment, $\mathcal{S}$ plays the role of $\mathcal{F}_{P_{V,W|X,Y}}$ and thus it can send $\mathcal{P}_2$'s randomness r_2 and valid y^i and w^i to $\mathcal{A}$.

– If $\mathcal{P}_2$ is corrupted after the decommitment, $\mathcal{S}$ also learns b and can send also b to $\mathcal{A}$.

We analyze below the probabilities of the events that can result in different views in the real execution of the protocol, with adversary $\mathcal{A}$, and in the ideal execution of the protocol, with simulator $\mathcal{S}$:

– The procedure of commitment with uncorrupted committer perfectly emulates the hybrid execution for the adversary $\mathcal{A}$.

– The procedure of decommitment with uncorrupted committer fails to emulate the hybrid execution for the adversary $\mathcal{A}$ if the simulator cannot equivocate, or if, in the hybrid interaction, an honest committer is unable to open a valid commitment. However, by lemma 2 the simulator can equivocate. And by the *δ-sound-and-binding* property of the *stand-alone secure*, the probability that an honest committer in the hybrid interaction is unable to open a valid commitment is negligible.

– The procedure of commitment with corrupted committer perfectly emulates the hybrid execution for the adversary $\mathcal{A}$ if the simulator successfully extracts b. Lemma 1 guarantees that the simulator can extract the bit b.

– The procedure of decommitment with corrupted committer fails to emulate the hybrid execution for the adversary $\mathcal{A}$ only if a dishonest committer in the hybrid interaction succeeds to open a bit other than the bit he committed to in the commitment phase. But by the *δ-sound-and-binding* property of the *stand-alone secure* protocol this probability is negligible.

– The procedure of corrupting $\mathcal{P}_1$ fails to emulate the hybrid execution for the adversary $\mathcal{A}$ only if the simulator cannot equivocate or cannot extract the value of the commitment (only in the case that the corruption occur during the commitment phase). But lemmas 1 and 2 guarantees that the simulator can extract the value of the commitment and can equivocate.

– The procedure of corrupting $\mathcal{P}_2$ perfectly emulates the hybrid execution for the adversary $\mathcal{A}$.

– A dishonest verifier that knows y^t, w^t, r_2, k in the hybrid interaction has, before the decommitment, negligible information about b according to the *ϵ-concealing* property of the *stand-alone secure* protocol.

We conclude that since all events that can result in different views have negligible probabilities, the protocol π UC-realizes $\mathcal{F}_{COM}$, and so the theorem is valid. □

5 Conclusion

In this paper, we prove commitment protocols based on two-party stateless functionalities matching a previously used ad-hoc list of security properties (binding, concealing and sound) are universally composable when unbounded simulators are allowed. As previously commented, this assumption on the simulator gives us secure universal composability with other statistically secure protocols, but it does not hold anymore when composing with computationally secure protocols. However, we note here that in the case there exists efficient procedures for extracting the commitment from $\mathcal{P}_1$'s messages and to find equivocations for a certain view, then our simulator becomes efficient. In those cases, we obtain the full generality of the UC composability theorem.

References

[1] M. Backes, J. Müller-Quade, D. Unruh. On the Necessity of Rewinding in Secure Multiparty Computation. *in Theory of Cryptography, Proceedings of TCC 2007*, Lecture Notes in Computer Science vol. 4392, Springer-Verlag, pp. 157-174, March 2007. Preprint on IACR ePrint 2006/315.

[2] B. Barak, R. Canetti, J. B. Nielsen, R. Pass. Universally Composable Protocols with Relaxed Set-Up Assumptions. *36th FOCS*, pp.186-195. 2004.

[3] G. Brassard, D. Chaum, C. Crépeau. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pages 156-189, 1988.

[4] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Available at http://eprint.iacr.org/2000/067. 2005. Extended Abstract appeared in proceedings of the *42nd Symposium on Foundations of Computer Science (FOCS)*, 2001.

[5] R. Canetti and M. Fischlin. Universally composable commitments. *In Advances in Cryptology - Crypto 2001*, pages 19-40, Berlin, 2001. Springer-Verlag. Lecture Notes in Computer Science Volume 2139.

[6] R. Canetti, Y. Lindell, R. Ostrovsky, A. Sahai. Universally Composable Two Party and Multiparty Secure Computation. *34th STOC*, pp. 494-503, 2002.

[7] C. Crépeau, J. Kilian. Achieving Oblivious Transfer using Weakened Security Assumptions. *Proc.* 29$^{\text{th}}$ *FOCS*, pp. 42–52. 1988.

[8] C. Crépeau. Efficient Cryptographic Protocols Based on Noisy Channels. *Advances in Cryptology: Proceedings of Eurocrypt '97* , Springer-Verlag, pages 306-317, 1997.

[9] C. Crépeau, G. Savvides, C. Schaffner, J. Wullschleger. Information-theoretic conditions for two-party secure function evaluation *Advances in Cryptology - EUROCRYPT '06*, LNCS, Springer-Verlag, 2006.

[10] C. Crépeau, J. Wullschleger. Statistical Security Conditions for Two-Party Secure Function Evaluation *Proceedings of ICITS 2008*, LNCS, Springer-Verlag, 2008.

[11] I. Damgård, J. Groth. Non interactive and reusable non-malleable commitment schemes. *34th STOC*, pp. 426-437. 2003.

[12] I. Damgård, J. Kilian, L. Salvail. On the (Im)possibility of Basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions. *Advances in Cryptology: EUROCRYPT 1999*, pp. 56–73. 1999.

[13] I. Damgård and J. B. Nielsen. Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor. *CRYPTO 2002*, pp.581-596. 2002.

[14] Y. Dodis, R. Pass and S. Walfish. Fully Simulatable Multiparty Computation. Manuscript, 2005.

[15] S. Even, O. Goldreich, A. Lempel. A Randomized Protocol for Signing Contracts. *Communications of the ACM 28(6)*, pp.637-647. 1985.

[16] O. Goldreich, Foundations of Cryptography : Volume 1 - Basic Tools. Cambridge University Press. 2001.

[17] O. Goldreich, S. Micali and A. Wigderson. How to Play Any Mental Game or a Completeness Theorem for Protocols with Honest Majority. *STOC '87*. 1987.

[18] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing but their Validity or All Languages in NP have Zero-Knowledge Proof System. *J. ACM 38(3)*: 691-729. 1991.

[19] I. Haitner, O. Horvitz, J. Katz, C. Koo, R. Morselli, R. Shaltiel: Reducing Complexity Assumptions for Statistically-Hiding Commitment. EUROCRYPT 2005: 58-77. The full version will appear in Journal of Cryptology.

[20] D. Hofheinz and J. Müller-Quade. Universally Composable Commitments Using Random Oracles. *Theory of Cryptography Conference (TCC)*, LNCS 2951 pp. 58-74. 2004.

[21] H. Imai, A. C. A. Nascimento, A. Winter. Commitment Capacity of Discrete Memoryless Channels. *IMA Int. Conf. 2003* pp.35-51. 2003.

[22] J. Katz. Universally Composable Multi-party Computation Using Tamper-Proof Hardware. *EUROCRYPT 2007*. pp. 115–128. 2007.

[23] E. Kushilevitz, Y. Lindell, T. Rabin. Information Theoretically Secure Protocols and Security under Composition. *STOC06*. 2006

[24] T. Moran and G. Segev. David and Goliath Commitments: UC Computation for Asymmetric Parties Using Tamper-Proof Hardware. *EUROCRYPT 2008*. pp. 527–544. 2008.

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