On the Composability of Statistically Secure Bit Commitments

Rafael Dowsley[1], Jeroen van de Graaf[2], Jörn Müller-Quade[3], and Anderson C. A. Nascimento[1]

[1] Department of Electrical Engineering, University of Brasilia,
Campus Darcy Ribeiro, 70910-900, Brasilia, DF, Brazil.
Email: rafaeldowsley@redes.unb.br, andclay@ene.unb.br
[2] Department of Computer Science, Federal University of Ouro Preto.
Ouro Preto, Minas Gerais, CEP: 35400-000, Brazil.
Email: jvdg@iceb.ufop.br
[3] Institut für Algorithmen und Kognitive Systeme, Universität Karlsruhe.
Am Fasanengarten 5, 76128 Karlsruhe, Germany.
Email: muellerq@ira.uka.de

Abstract. We show that for bit commitment schemes based on two-party stateless primitives, the stand-alone statistical security implies the statistical universally composable security. I.e. all such schemes are secure with an unlimited adversary, an unlimited simulator and an unlimited environment machine in the universal composability framework.
Especially, these protocols can be used in arbitrary statistically secure applications without lowering the security.
Keywords: Commitment, Universal Composability, Two-party Stateless Primitives, Noisy Channels.

1 Introduction

Commitment Protocols: Commitment is one of the most fundamental cryptographic primitives. It is used as a sub-protocol in applications such as secure multi-party computation [19], contract signing [17] and zero-knowledge proofs [20, 18, 4]. A commitment protocol involves two players: the committer and the verifier. The idea behind a commitment is simple: the committer provides the verifier with a digital equivalent of a "sealed envelope" in the commitment phase of the protocol. This envelope should contain a value x. The verifier should be unable to learn anything about the value x before the committer helps him to open the committed value. Additionally, the committer should not be able to change x after the commitment phase. When the committer helps the verifier to open the committed value (in the decommitment phase), the verifier should learn the value x.
Universal Composability: A very large number of commitment protocols, based on various assumptions, are known in the stand-alone setting. 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 [5, 6] is a notion of security that guarantees the security of the scheme even when it 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 setup assumption is assumed [6], however it is possible to achieve such notion of security based on some setup assumption. One of the setup assumptions with which UC commitment schemes were constructed is the common reference string (CRS) model [6, 7, 14, 12]. In the CRS model there exists a random string that is honestly generated at the system initialization and that is available to all parties. In this model, the simulator can generate its own string (as long as it looks indistinguishable from the honestly generated one). Barak et al. [2] presented UC secure commitment schemes that work in the PKI model in the presence of a static adversary.

In this model the parties have certified public keys. Dodis et al. [15] extend these results to adaptive corruptions. It is also possible to build a UC commitment procotol in the random oracle model [22]. Prabhakaran and Sahai [27] 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 [24, 26] UC Commitment protocols based on tamper-proof hardware were proposed. Finally, Dowsley et al. [16] showed that it is possible to obtain UC Commitment in the noisy channels model.

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 [25, 1]. In general, these securities notions are not equivalent [1]. Therefore, it is an interesting question to study if there are restricted scenarios in which 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 notion of security, but actually are universally composable. As a particular consequence of our result, all the previously published commitment schemes based on noisy channels [9, 8, 13, 23] are actually secure when arbitrarily composed (assuming that the noisy channels behave as ideal functionalities). Our result also implies the UC security of the commitment schemes based on pre-distributed data [3, 28].

Related Work: The question about the composability of statistically secure protocols has been previously addressed in [25, 1], where it was proven that the equivalence does not hold in general. In [10, 11] 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 [11]. The parallel composability of statistically-hiding commitments was shown in [21].

2 The UC Framework

This section briefly reviews the main concepts of the UC framework and describes the ideal functionalities that we use in this work. We refer the reader to [5] for a more detailed explanation of this framework.

2.1 Notation

We will denote by calligraphic letters the domains of the random variables and other sets, by $|\mathcal{X}|$ the cardinality of a set $\mathcal{X}$, by upper case letters the random variables and by lower case letters one realization of the random variable. We also use calligraphic letters for the environment, the adversary, the simulator, the ideal functionalities and the parties. For a random variable X over $\mathcal{X}$, we denote its probability distribution by $P_X : \mathcal{X} \to [0, 1]$ with $\sum_{x\in\mathcal{X}} P_X(x) = 1$. For a joint probability distribution $P_{XY} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$, let $P_X(x) := \sum_{y\in\mathcal{Y}} P_{XY}(x, y)$ denote the marginal probability distribution and let $P_{X|Y}(x|y) := \frac{P_{XY}(x,y)}{P_Y(y)}$ denote the conditional probability distribution if $P_Y(y) \neq 0$.

The statistical distance between two probability distributions P_X and P_Y over the same domain $\mathcal{V}$ is

$$\mathsf{SD}(P_X, P_Y) := \frac{1}{2}\sum_{v\in\mathcal{V}} |P_X(v) - P_Y(v)|.$$

2.2 Overview

In the UC framework, the security of a protocol to carry out a certain task is established 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 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. The ideal adversary, a.k.a. simulator, communicates with the trusted party.
3. One proves that the real protocol emulates the ideal protocol. I.e. one shows that for every adversary in the real model there is a simulator interacting in the the ideal model and such that no environment machine can distinguish if it is interacting with the real or with the ideal execution.

The environment machine represents all activities that are external to the analyzed protocol. Hence 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 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. When one replaces the ideal functionalities used in a protocol by sub-protocols that UC-emulates these functionalities, the security is retained.

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 $\mathcal{Z}$ with 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 except that provided by the ideal functionality. The environment machine, $\mathcal{Z}$, can set the inputs to the parties, and read their outputs, but cannot see their communication with the ideal functionality.

The real protocol. In the real world, the protocol π is executed by parties $\mathcal{P}_1, \ldots, \mathcal{P}_n$ 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 cannot see 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 $\mathcal{A}$. 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 corrupting process, 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, 25, 11, 10].

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 the protocol π is statistically indistinguishable from interacting with an ideal protocol for $\mathcal{F}$.

2.3 The Commitment Functionality

We present the ideal bit commitment functionality as described in [5] (it is a modified version of the first formalized functionality [6]). 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 $sid = (\mathcal{P}_1, \mathcal{P}_2, sid')$ 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 sends a public delayed output to the verifier (i.e., the message is first sent to the adversary, and then sent to the verifier whenever the adversary allows it) to notify that a value was committed to. 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 in 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. $\mathcal{F}_{COM}$ 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.

2.4 Statistically Secure Two Party Stateless Functionality

We introduce an hybrid model in which the participants have 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 $\mathcal{Y}$. Then the functionality produces outputs 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 $sid = (\mathcal{P}_1, \mathcal{P}_2, sid')$ for some $\mathcal{P}_2$ and that $x \in \mathcal{X}$, else ignore the input. If there is already a recorded value with this *sid* ignore the input, otherwise record *sid* and the corresponding x. If the values x and y corresponding to *sid* are already recorded, choose randomly v, w according to $P_{V,W|X,Y}$ and 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 $sid = (\mathcal{P}_1, \mathcal{P}_2, sid')$ for some $\mathcal{P}_1$ and that $y \in \mathcal{Y}$, else ignore the input. If there is already a recorded value with this *sid* ignore the input, otherwise record *sid* and the corresponding y. If the values x and y corresponding to *sid* are already recorded, choose randomly v, w according to $P_{V,W|X,Y}$ and output v to $\mathcal{P}_1$ and w to $\mathcal{P}_2$.

This functionality models a noisy channel when we set $\mathcal{Y}$ and $\mathcal{V}$ to be unary alphabets. It can also model pre-distributed data by setting $\mathcal{X}$ and $\mathcal{Y}$ to be unary alphabets (i.e., the outputs do not depend on parties' actions).
In the ideal process for the functionality $\mathcal{F}_{COM}$ (described in section 2.3) $\mathcal{F}_{P_{V,W|X,Y}}$ is not used. Therefore a simulator $\mathcal{S}$, that internally executes 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 the commitment protocols based on 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 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 inputs/outputs until round i. At the end of this commitment phase, $\mathcal{P}_2$ outputs (RECEIPT, *sid*). Note that this model 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 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 model 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.
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 the random variable $View_2$. $View_1$ is defined similarly for the party $\mathcal{P}_1$.

We will now define the security of a protocol. A protocol is δ-*concealing* if for any possible behavior of $\mathcal{P}_2$ in the commitment phase when interacting with an honest committer, we have that

$$\mathsf{SD}(P_{View_2|B=0}, P_{View_2|B=1}) \leq \delta$$

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

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

where the probability is taken over the randomness of the parties and of the functionality.
Consider a protocol execution between an honest receiver and a (possibly malicious) committer $\mathcal{P}_1$. We call the protocol δ-*binding* if for every possible algorithm $\mathcal{P}_1$ that interact with the honest receiver in the commitment phase and then outputs $((\widetilde{r_1}, \widetilde{x^t}, \widetilde{v^t}, 0), (\widehat{r_1}, \widehat{x^t}, \widehat{v^t}, 1))$, we have that

$$\begin{aligned}\Pr[&\beta(\widetilde{X^t}, Y^t, \widetilde{V^t}, W^t, R_2, \widetilde{R_1}, K, 0) = accept \ \& \\ &\beta(\widehat{X^t}, Y^t, \widehat{V^t}, W^t, R_2, \widehat{R_1}, K, 1) = accept] \leq \delta.\end{aligned}$$

where the probability is taken over the randomness of the parties and of the functionality.
We call the protocol *stand-alone secure* if δ is 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 will be used 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 the inputs and outputs of the functionality and the view of $\mathcal{P}_2$ in the commitment phase, one can almost surely infer $\mathcal{P}_1$'s commitment (this property is known as extractability).

Definition 1 (Extractability). *A commitment scheme is extractable if there exists an algorithm which given*
1. *the inputs/outputs of the stateless functionality*
2. *the view of the honest receiver after the commit phase*

computes with overwhelming probability (not necessarily efficiently) the value that was committed to.

Lemma 1. *Any stand-alone secure protocol of commitment based on two party stateless functionality supports the extraction of the committed value.*

The algorithm is given all the inputs/outputs to the two-party stateless functionality and also $\mathcal{P}_2$ view after the commit phase, so it knows x^t, v^t and k. The algorithm uses this knowledge to computes the randomness $\widetilde{R_1}$ and the bit $\widetilde{b}$ that maximize the probability that $\beta(X^t, Y^t, V^t, W^t, R_2, \widetilde{R_1}, K, \widetilde{b}) = accept$, breaking ties in some arbitrary way.
By the bindingness we get

$$\begin{aligned}\Pr[&\beta(X^t, Y^t, V^t, W^t, R_2, \widetilde{R_1}, K, \widetilde{b}) = accept \ \& \\ &\beta(X^t, Y^t, V^t, W^t, R_2, \widehat{R_1}, K, 1-\widetilde{b}) = accept] \leq \delta.\end{aligned}$$

for any $\widetilde{R_1}, \widehat{R_1}$ that is independent of R_2, Y^t, W^t when conditioned on $View_1$. Hence, there is at most one committed value such that the test succeeds with overwhelming probability over the randomness of the receiver and of the functionality. In the case that such value exists, it is equal to $\widetilde{b}$ with overwhelming probability, since we picked the value $\widetilde{b}$ that maximized the success probability.
We now prove that, for any possible view of $\mathcal{P}_2$ after the commitment phase and for any possible committed value b', there exists at least one view of the honest committer $\mathcal{P}_1$ that passes the test executed by $\mathcal{P}_2$ at the end of the decommitment phase (the so called equivocability property of the commitment protocol).

Definition 2 (Equivocability). *A commitment scheme has the equivocability property if given any possible view of $\mathcal{P}_2$ after the commitment phase and for any value b', it is possible to compute, with overwhelming probability of suscess (not necessarily efficiently), a valid view for the honest committer $\mathcal{P}_1$ such that $View_1$ passes the test executed in the decommitment phase and opens the commitment as b'.*

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

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

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

where the probability is taken over the randomness of the parties and of the functionality. We claim that there exist $\widetilde{x^t}, \widetilde{v^t}, \widetilde{r_1}$, such that

$$\beta(\widetilde{x^t}, y^t, \widetilde{v^t}, w^t, r_2, \widetilde{r_1}, k, 1-b) = accept$$

because if this were not the case, $\mathcal{P}_2$ could break the concealing condition by just computing, for all values $\widehat{x^t}, \widehat{v^t}, \widehat{r_1}$ and $\widehat{b}$,

$$\beta(\widehat{x^t}, y^t, \widehat{v^t}, w^t, r_2, \widehat{r_1}, k, \widehat{b})$$

Then no tuple of values will make an honest receiver accept the value $1-b$, but, with overwhelming probability, there will be a tuple of values such that an honest receiver would accept b. Therefore, given only the view of the receiver, it is possible to break the concealing condition.
We now use lemmas 1 and 2 to prove our main result, which is stated below:

Theorem 1. *Any stand-alone secure protocol of commitment based on two party stateless functionality statistically UC-realize $\mathcal{F}_{COM}$ using $\mathcal{F}_{AUTH}$ and $\mathcal{F}_{P_{V,W|X,Y}}$.*

We construct the simulator $\mathcal{S}$ 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 $\mathcal{A}$. In addition, $\mathcal{S}$ forwards all the communication between $\mathcal{Z}$ and $\mathcal{A}$.
Basically, $\mathcal{S}$ should be able to: (1) extract the committed value from the messages that it receives from $\mathcal{A}$, if $\mathcal{P}_1$ is corrupted. (2) send a commitment in the hybrid interaction and later open it to any value, if $\mathcal{P}_1$ is honest. 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 $\mathcal{S}$ receives the public delayed output (RECEIPT, *sid*) from $\mathcal{F}_{COM}$. $\mathcal{S}$ chooses a random bit b' to commit to in the simulated 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 then simulates the real execution of the protocol using a simulated $\mathcal{F}_{P_{V,W|X,Y}}$. $\mathcal{S}$ reveals to $\mathcal{A}$ only the noiseless messages generated in this simulated execution). When $\mathcal{A}$ delivers all noiseless messages in the simulated interaction, $\mathcal{S}$ 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 noiseless messages of an honest committer in each round (i.e., $\mathcal{S}$ chooses the randomness r_1 of the simulated committer at the beginning of the first round. Then $\mathcal{S}$ simulates the noiseless messages generated by the simulated committer in each round, using the view of the simulated committer at each point to determine the next message). Whenever $\mathcal{A}$ delivers all the noiseless messages in round i and sends the input $y_i \in \mathcal{Y}$ of the corrupted verifier to $\mathcal{F}_{P_{V,W|X,Y}}$, $\mathcal{S}$, then $\mathcal{S}$ simulates the outputs of that functionality according to the probability $P_{V,W|X,Y}$ (using $x_i \in \mathcal{X}$ that is determined by the view of the simulated committer at that point) and sends w_i to $\mathcal{A}$. 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 public delayed output (OPEN, *sid*, x) from $\mathcal{F}_{COM}$. By lemma 2, $\mathcal{S}$ can find (with overwhelming probability) $\tilde{x}^t$, $\tilde{v}^t$ and $\tilde{r_1}$ such that after the exchange of messages in the decommitment phase, the bit b will be accepted by an honest verifier. $\mathcal{S}$ 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 in the simulated interaction.
- 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. Then $\mathcal{S}$ proceeds as follows:

- If $\mathcal{P}_2$ is uncorrupted, $\mathcal{S}$ simulates for $\mathcal{A}$ the noiseless messages of an honest verifier in each round (i.e., $\mathcal{S}$ chooses the randomness r_2 of the simulated verifier at the beginning of the first round. Then $\mathcal{S}$ simulates the noiseless messages generated by the simulated verifier in each round by using the view of the simulated verifier at each point to determine the next message). Whenever $\mathcal{A}$ delivers all the noiseless messages in round i and sends the input $x_i \in \mathcal{X}$ of the corrupted committer to $\mathcal{F}_{P_{V,W|X,Y}}$, $\mathcal{S}$ simulates the outputs of that functionality according to the probability $P_{V,W|X,Y}$ (using $y_i \in \mathcal{Y}$ that is determined by the view of the simulated verifier at that point) and sends v_i to $\mathcal{A}$. The simulator proceeds to the next round. When all t rounds finish, if the simulated verifier accept 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 $\mathcal{F}_{P_{V,W|X,Y}}$ for the corrupted parties in the simulated hybrid execution.

Decommitment - Corrupted Committer: If $\mathcal{A}$ tells some corrupted $\mathcal{P}_1$ to open a commitment with bit b' and $\mathcal{P}_2$ is uncorrupted, $\mathcal{S}$ simulates for $\mathcal{A}$ the messages sent by the honest verifier in the decommitment phase of the hybrid interaction. $\mathcal{S}$ then checks b' following the procedures used by an honest verifier. 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 the committer, then $\mathcal{S}$ corrupts $\mathcal{P}_1$ 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 of the commitment phase in the simulated hybrid execution and thus $\mathcal{A}$ 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. Then $\mathcal{S}$ follows the same procedures as in the case of commitment with corrupted committer in the remaining rounds, extracts the new value b' and sends it to $\mathcal{F}_{COM}$.
- If the (RECEIPT, *sid*) output was written on $\mathcal{P}_2$'s tape before the corruption, $\mathcal{A}$ knows k, and possibly y^t, w^t and r_2 (the last three only if $\mathcal{P}_2$ is already corrupted). Then $\mathcal{S}$ 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 the verifier, then $\mathcal{S}$ corrupts $\mathcal{P}_2$ in the ideal protocol. $\mathcal{S}$ proceeds as follows:

- If the verifier is corrupted in the simulated hybrid execution after round i of the commitment phase 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 the verifier is corrupted after the decommitment, $\mathcal{S}$ also learns b and thus can also send b to $\mathcal{A}$.

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

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

- The procedure of decommitment with an uncorrupted committer fails to emulate the hybrid execution for the adversary $\mathcal{A}$ only 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 with overwhelming probability. And by the *δ-correct* property of the *stand-alone security*, the probability that an honest committer 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 with overwhelming probability.
- The procedure of decommitment with corrupted committer fails to emulate the hybrid execution for the adversary $\mathcal{A}$ only if a dishonest committer succeeds, in the hybrid interaction, to open a bit other than the bit he committed to. But by the *δ-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 new value of the commitment (in the case that the corruption occurs during the commitment phase). But lemmas 1 and 2 guarantee that with overwhelming probability 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 *δ-concealing* property of the *stand-alone secure* protocol.

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 proved that commitment protocols based on two-party stateless functionalities matching a previously used ad-hoc list of security properties (binding, concealing and correctness) 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 if there exist 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 and therefore we obtain the full generality of the UC composability theorem for the analyzed protocol.

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. D. Beaver. Commodity-Based Cryptography (Extended Abstract). *STOC* 1997: 446-455.
4. G. Brassard, D. Chaum, C. Crépeau. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pages 156-189, 1988.
5. 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.

6. 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.
7. R. Canetti, Y. Lindell, R. Ostrovsky, A. Sahai. Universally Composable Two Party and Multi-party Secure Computation. *34th STOC*, pp. 494-503, 2002.
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, J. Kilian. Achieving Oblivious Transfer using Weakened Security Assumptions. *Proc.* 29^{th} *FOCS*, pp. 42–52. 1988.
10. 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.
11. C. Crépeau, J. Wullschleger. Statistical Security Conditions for Two-Party Secure Function Evaluation *Proceedings of ICITS 2008*, LNCS, Springer-Verlag, 2008.
12. I. Damgård, J. Groth. Non interactive and reusable non-malleable commitment schemes. *34th STOC*, pp. 426-437. 2003.
13. 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.
14. 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.
15. Y. Dodis, R. Pass and S. Walfish. Fully Simulatable Multiparty Computation. Manuscript, 2005.
16. R. Dowsley, J. Müller-Quade, A. C. A. Nascimento. On Possibility of Universally Composable Commitments Based on Noisy Channels. *SBSEG 2008*, pp. 103–114. 2008.
17. S. Even, O. Goldreich, A. Lempel. A Randomized Protocol for Signing Contracts. *Communications of the ACM 28(6)*, pp.637-647. 1985.
18. O. Goldreich, Foundations of Cryptography : Volume 1 - Basic Tools. Cambridge University Press. 2001.
19. 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.
20. 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.
21. 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.
22. D. Hofheinz and J. Müller-Quade. Universally Composable Commitments Using Random Oracles. *Theory of Cryptography Conference (TCC)*, LNCS 2951 pp. 58-74. 2004.
23. H. Imai, A. C. A. Nascimento, A. Winter. Commitment Capacity of Discrete Memoryless Channels. *IMA Int. Conf. 2003* pp.35-51. 2003.
24. J. Katz. Universally Composable Multi-party Computation Using Tamper-Proof Hardware. *EUROCRYPT 2007*. pp. 115–128. 2007.
25. E. Kushilevitz, Y. Lindell, T. Rabin. Information Theoretically Secure Protocols and Security under Composition. *STOC06*. 2006
26. T. Moran and G. Segev. David and Goliath Commitments: UC Computation for Asymmetric Parties Using Tamper-Proof Hardware. *EUROCRYPT 2008*. pp. 527–544. 2008.
27. M. Prabhakaran and A. Sahai. New Notions of Security: Achieving Universal Composability without Trusted Setup. In *Proc. of STOC*, 2004.
28. R. L. Rivest. Unconditionally Secure Commitment and Oblivious Transfer Schemes Using Private Channels and a Trusted Initializer. Manuscript. 1999.