Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor

Ivan Damgård and Jesper Buus Nielsen[*]

BRICS[**] Department of Computer Science
University of Aarhus
Ny Munkegade
DK-8000 Arhus C, Denmark

Abstract. Canetti and Fischlin have recently proposed the security notion *universal composability* for commitment schemes and provided two examples. This new notion is very strong. It guarantees that security is maintained even when an unbounded number of copies of the scheme are running concurrently, also it guarantees non-malleability and security against adaptive adversaries. Both proposed schemes use $\Theta(k)$ bits to commit to one bit and can be based on the existence of trapdoor commitments and non-malleable encryption.
We present new universally composable commitment (UCC) schemes based on extractable q one-way homomorphisms. These in turn exist based on the Paillier cryptosystem, the Okamoto-Uchiyama cryptosystem, or the DDH assumption. The schemes are efficient: to commit to k bits, they use a constant number of modular exponentiations and communicates $O(k)$ bits. Furthermore the scheme can be instantiated in either perfectly hiding or perfectly binding versions. These are the first schemes to show that constant expansion factor, perfect hiding, and perfect binding can be obtained for universally composable commitments.
We also show how the schemes can be applied to do efficient zero-knowledge proofs of knowledge that are universally composable.

1 Introduction

The notion of commitment is one of the most fundamental primitives in both theory and practice of modern cryptography. In a commitment scheme, a *committer* chooses an element m from some finite set M, and releases some information about m through a commit protocol to a *receiver*. Later, the committer may release more information to the receiver to open his commitment, so that the receiver learns m. Loosely speaking, the basic properties we want are first that the commitment scheme is *hiding:* a cheating receiver cannot learn m from

[*] {ivan,buus}@brics.dk.

[**] Basic Research in Computer Science,
Centre of the Danish National Research Foundation.

the commitment protocol, and second that it is *binding:* a cheating committer cannot change his mind about m, the verifier can check in the opening that the value opened was what the committer had in mind originally. Each of the two properties can be satisfied unconditionally or relative to a complexity assumption. A very large number of commitment schemes are known based on various notions of security and various complexity assumptions.

In [**?**] Canetti and Fischlin proposed a new security measure for commitment schemes called universally composable commitments. This is a very strong notion: it guarantees that security is maintained even when an unbounded number of copies of the scheme are running concurrently and asynchronous. It also guarantees non-malleability and maintains security even if an adversary can decide adaptively to corrupt some of the players and make them cheat. The new security notion is based on the framework for universally composable security in [**?**]. In this framework one specifies desired functionalities by specifying an idealized version of them. An idealized commitment scheme is modeled by assuming a trusted party to which both the committer and the receiver have a secure channel. To commit to m, the committer simply sends m to the trusted party who notifies the receiver that a commitment has been made. To open, the committer asks the trusted party to reveal m to the receiver. Security of a commitment scheme now means that the view of an adversary attacking the scheme can be simulated given access to just the idealized functionality.

It is clearly important for practical applications to have solutions where only the two main players need to be active. However, in [**?**] it is shown that universal composability is so strong a notion that no universally composable commitment scheme for only two players exist. However, if one assumes that a common reference string with a prescribed distribution is available to the players, then two-player solutions do exist and two examples are given in [**?**]. Note that common reference strings are often available in practice, for instance if a public key infrastructure is given.

The commitment scheme(s) from [**?**] uses $\Omega(k)$ bits to commit to one bit, where k is a security parameter, and it guarantees only computational hiding and binding. In fact, as detailed later, one might even get the impression from the construction that perfect hiding, respectively binding cannot be achieved. Here, by perfect, we mean that an unbounded receiver gets zero information about m, respectively an unbounded committer can change his mind about m with probability zero.

Our contribution is a new construction of universally composable commitment schemes, which uses $O(k)$ bits of communication to commit to k bits. The scheme can be set up such that it is perfectly binding, or perfectly hiding, without loosing efficiency[1]. The construction is based on a new primitive which we call a mixed commitment scheme. We give a general construction of mixed

[1] [**?**] also contains a scheme which is statistically binding and computationally hiding, the scheme however requires a new setup of the common reference string per commitment and is thus mostly interesting because it demonstrates that statistically binding can be obtained at all.

commitments, based on any family of so called extractable q one-way homomorphisms, and show two efficient implementations of this primitive, one based on the Paillier cryptosystem and one based on the Okamoto-Uchiyama cryptosystem. A third example based on the DDH assumption is less efficient, but still supports perfect hiding or binding. Our commitment protocol has three moves, but the two first messages can be computed independently of the message committed to and thus the latency of a commitment is still one round as in [**?**]. We use a "personalized" version of the common reference string model where each player has a separate piece of the reference string assigned to him. It is an open question if our results can also be obtained with a reference string of size independent of the number of players.

As a final contribution we show that if a mixed commitment scheme comes with protocols in a standard 3-move form for proving in zero-knowledge relations among committed values, the resulting UCC commitment scheme inherits these protocols, such that usage of these is also universally composable. For our concrete schemes, this results in efficient protocols for proving binary Boolean relations among committed values and also (for the version based on Paillier encryption) additive and multiplicative relations modulo N. We discuss how this can be used to construct efficient universally composable zero-knowledge proofs of knowledge for NP, improving the complexity of a corresponding protocol from [**?**].

An Intuitive Explanation of some Main Ideas In the simplest type of commitment scheme, both committing and opening are non-interactive, so that committing just consists of running an algorithm commit_K, keyed by a public key K, taking as input the message m to be committed to and a uniformly random string r. The committer computes $c \leftarrow \text{commit}_K(m, r)$, and sends c to the receiver. To open, the committer sends m and r to the receiver, who checks that $c = \text{commit}_K(m, r)$. For this type of scheme, hiding means that given just c the receiver does not learn m and binding means that the committer cannot change his mind by computing m', r', where $c = \text{commit}(m', r')$ and $m' \neq m$.

In a *trapdoor scheme* however, to each public key K a piece of trapdoor information t_K is associated which, if known, allows the committer to change his mind. We will call such schemes *equivocable.* One may also construct schemes where a different type of trapdoor information d_K exists, such that given d_K, one can efficiently compute m from $\text{commit}_K(m, r)$. We call such schemes *extractable.* Note that equivocable schemes cannot be perfect binding and that extractable schemes cannot be perfect hiding.

As mentioned, the scheme in [**?**] guarantees only computational binding and computational hiding. Actually this is important to the construction: to prove security, we must simulate an adversary's view of the real scheme with access to the idealized functionality only. Now, if the committer is corrupted by the adversary and sends a commitment c, the simulator must find out which message was committed to, and send it to the idealized functionality. The universally composable framework makes very strict demands to the simulation implying that rewinding techniques cannot be used for extracting the message. A solution is to use an extractable scheme, have the public key K in the reference string, and set things up such that the simulator knows the trapdoor d_k. A similar consideration leads to the conclusion that if instead the receiver is corrupt, the scheme must be equivocable with trapdoor t_K known to the simulator, because the simulator must generate a commitment on behalf of the honest committer before finding out from the idealized functionality which value was actually committed to. So to build universally composable commitments it seems we must have a scheme that is simultaneously extractable *and* equivocable. This is precisely what Canetti's and Fischlin's ingenious construction provides.

In this paper, we propose a different technique for universally composable commitments based on what we call a mixed commitment scheme. A mixed commitment scheme is basically a commitment scheme which on some of the keys is perfectly hiding and equivocable, we call these keys the E-keys, and on some of the keys is perfectly binding and extractable, we call these keys the X-keys. Clearly, no key can be both an X- and an E-key, so if we were to put the entire key in the common reference string, either extractability or equivocability would fail and the simulation could not work. We remedy this by putting only a part of the key, the so-called system key, in the reference string. The rest of the key is set up once per commitment using a two-move protocol. This allows the simulator to force the key used for each commitment to be an E-key or an X-key depending on whether equivocability or extractability is needed.

Our basic construction is neither perfectly binding nor perfectly hiding because the set-up of keys is randomized and is not guaranteed to lead to any particular type of key. However, one may add to the reference string an extra key that is guaranteed to be either an E- or an X-key. Using this in combination with the basic scheme, one can obtain either perfect hiding or perfect binding.

2 Mixed Commitments

We now give a more formal description of mixed commitment schemes. The most important difference to the intuitive discussion above is that the system key N comes with a trapdoor t_N that allows efficient extraction for all X-keys. The E-keys, however, each come with their own trapdoor for equivocability.

Definition 1. *By a* mixed commitment scheme *we mean a commitment scheme* commit_K *with some global system key N, which determines the message space $\mathcal{M}_N$ and the key space $\mathcal{K}_N$ of the commitments. The key space contains two sets, the E-keys and the X-keys, for which the following holds:*

Key generation *One can efficiently generate a system key N along with the so-called X-trapdoor t_N. One can, given the system key N, efficiently generate random keys from $\mathcal{K}_N$ and given t_N one can sample random X-keys. Given the system key, one can efficiently generate an E-key K along with the so-called E-trapdoor t_K.*

Key indistinguishability *Random E-keys and random X-keys are both computationally indistinguishable from random keys from $\mathcal{K}_N$ as long as the X-trapdoor t_N is not known.*

Equivocability *Given E-key K and E-trapdoor t_K one can generate fake commitments c, distributed exactly as real commitments, which can later be opened arbitrarily, i.e. given a message m one can compute uniformly random r for which $c = \mathrm{commit}_K(m, r)$.*

Extraction *Given a commitment $c = \mathrm{commit}_K(m, r)$, where K is an X-key, one can given the X-trapdoor t_N efficiently compute m.*

Note that the indistinguishability of random E-keys, random X-keys, and random keys from $\mathcal{K}_N$ implies that as long as the X-trapdoor is not known the scheme is computationally hiding for all keys and as long as the E-trapdoor is not known either the scheme is computationally binding for all keys.

For the construction in the next section we will need a few special requirements on the mixed commitment scheme. First of all we will assume that the message space $\mathcal{M}_N$ and the key space $\mathcal{K}_N$ are finite groups in which we can compute efficiently. We denote the group operation by $+$. Second we need that the number of E-keys over the total number of keys is negligible and that the number of X-keys over the total number of keys is negligibly close to 1. Note that this leaves only a negligible fraction which is neither X-keys nor E-keys. We call a mixed commitment scheme with these properties a special mixed commitment scheme.

The last requirement is that the scheme has two 'independent' E-trapdoors per E-key. We ensure this by a transformation. The keys will be of the form (K_1, K_2). We let the E-keys be the pairs of E-keys and let the X-keys be the pairs of X-keys. The message space will be the same. Given a message m we commit as $(\mathrm{commit}_{K_1}(\overline{m}_1), \mathrm{commit}_{K_2}(\overline{m}_2))$, where $\overline{m}_1$ and $\overline{m}_2$ are uniformly random values for which $m = \overline{m}_1 + \overline{m}_2$. If both keys are E-keys and the E-trapdoor of one of them, say K_b, is known a fake commitment is made by committing honestly to random m_{1-b} under K_{1-b} and making a fake commitment c_b under K_b. Then to open to m, open c_b to $m_b = m_{1-b} - m$. Note that the distribution of the result is independent of b — this will become essential later. All requirements for a special mixed commitment scheme are maintained under the transformation.

Special Mixed Commitment Scheme based on q One-Way Homomorphisms Our examples of special mixed commitment schemes are all based on q one-way homomorphism generators, as defined in [**?**]. Here we extend the notion to extractable q one-way homomorphisms. In a nutshell, we want to look at an easily computable homomorphism $f : G \to H$ between Abelian groups G, H such that $H/f(G)$ is cyclic and has only large prime factors in its order. And such that random elements in $f(G)$ are computationally indistinguishable from random elements chosen from all of H (which in particular implies that f is hard to invert). However, given also a trapdoor associated with f, it becomes easy to extract information about the status of an element in H.

More formally, a family of extractable q one-way homomorphisms is given by a probabilistic polynomial time (PPT) generator $\mathcal{G}$ which on input 1^k outputs a (description of a) tuple $(G, H, f, g, q, b, b', t)$, where G and H are groups, $f : G \to H$ is an efficiently computable homomorphism, $g \in H \setminus f(G)$, $q, b, b' \in \boldsymbol{N}$, and t is a string called the trapdoor. Let $F = f(G)$. We require that gF generates the factor group H/F and let $\mathrm{ord}(g) = |H/F|$. We require that $\mathrm{ord}(g)$ is superpolynomial in k (e.g. 2^k), that q is a multiple of $\mathrm{ord}(g)$, and that b is a public lower bound on $\mathrm{ord}(g)$, i.e., we require that $2 \le b \le \mathrm{ord}(g) \le q$. We say that a generator has *public order* if $b = \mathrm{ord}(g) = q$. Also b' is superpolynomial in k (e.g. $2^{k/2}$) and it is a public lower bound on the primefactors in $\mathrm{ord}(g)$, i.e., all primefactors in $\mathrm{ord}(g)$ are at least b'. We write operations in G and H multiplicatively and we require that in both groups one can multiply, exponentiate, take inverses, and sample random elements in PPT given (G, H, f, g, q, b, b'). The final central requirements are as follows:

Indistinguishability. Random elements from F are *computationally indistinguishable* from random elements from H given (G, H, f, g, q, b, b').

Extractability. This comes in two flavors. We call the generator *fully extractable* if given $(G, H, f, g, q, b, b', t)$ and $y = g^i f(r)$ one can compute $i \bmod \mathrm{ord}(g)$ in PPT. Note that, given $(G, H, f, g, q, b, b', t)$, one can compute $\mathrm{ord}(g)$ easily. We call a generator 0/1-*extractable* if given $(G, H, f, g, q, b, b', t)$ and $y = g^i f(r)$ one can determine whether $i = 0$ in PPT.

q-invertibility Given (G, H, f, g, q, b, b') and $y \in H$, it is easy to compute x such that $y^q = f(x)$. Note that this does not contradict indistinguishability: since q is a multiple of $\mathrm{ord}(g)$, it is always the case that $y^q \in F$.

We give three examples of extractable q one-way homomorphism generators:

Based on Paillier encryption: Let $n = PQ$ be an RSA modulus, where P and Q are $k/2$-bit primes. Let $G = \boldsymbol{Z}_n^*$, let $H = \boldsymbol{Z}_{n^2}^*$, and let $f(r) = r^n \bmod n^2$. Let $g = (n + 1)$, let $b = q = n$, $b' = 2^{k/2-1}$,and let $t = (P, Q)$. Then it follows directly from [**?**] that relative to the DCRA assumption we have a fully extractable generator with public order.

Based on Okamoto-Uchiyama encryption: Now let $N = Pn = P^2Q$. Let $G = \boldsymbol{Z}_n^*$, let $H = \boldsymbol{Z}_N^*$, and let $f(r) = r^N \bmod N$. Let $g = (N + 1)$, $q = N$, $b = b' = 2^{k/2-1}$ and $t = (P, Q)$. Then it follows directly from [**?**] that relative to the p-subgroup assumption we have a fully extractable generator.

Based on Diffie-Hellman encryption: Let $\langle\alpha\rangle$ be a group of prime order Q. Let $\beta = \alpha^x$ for uniformly random $x \in \boldsymbol{Z}_Q^*$. Let $G = \boldsymbol{Z}_Q$, let $H = \langle\alpha\rangle \times \langle\alpha\rangle$, and let $f(r) = (\alpha^r, \beta^r)$. Let $g = (1, \beta)$, $b = b' = q = Q$ and $t = x$. Then the scheme is 0/1-extractable: let $(A, B) = g^m f(r) = (\alpha^r, \beta^{r+m})$, then $A^x = B$ iff $m = 0$. Relative to the DDH assumption we have a 0/1-extractable generator with public order.

We now show how to transform an extractable generator into a special mixed commitment scheme. We treat fully extractable and 0/1-extractable generators in parallel, as the differences are minimal.

The key space will be H, the message space will be $\boldsymbol{Z}_b$ for fully extractable schemes and $\boldsymbol{Z}_2$ for 0/1-extractable schemes. We commit as $\mathrm{commit}_K(m, r) = K^m f(r)$, where r is uniformly random in H. The E-keys will be the set $F = f(G)$ and the E-trapdoor will be $f^{-1}(K)$. By the requirement that $\mathrm{ord}(g)$ is superpolynomial in k, the set of E-keys is a negligible fraction of the keyspace as required. For equivocability, we generate a fake commitment as $c = f(r_c)$ for uniformly random $r_c \in H$. Assume that $K = f(r_K)$ and that we are given $m \in \boldsymbol{Z}_b$. Compute $r = r_K^{-m} r_c$. Then r is uniformly random and $c = \mathrm{commit}_K(m, r)$.

For a fully extractable generator the X-keys will be the elements of form $K = g^i f(r_K)$, where i is invertible in $\boldsymbol{Z}_{\mathrm{ord}(g)}$. By the requirement that $\mathrm{ord}(g)$ only has large primefactors, the X-keys are the entire key-space except for a negligible fraction as required. They can be sampled efficiently given the trapdoor since then $\mathrm{ord}(g)$ is known. Assume that we are given $c = K^m f(r)$ for $m \in \boldsymbol{Z}_b$. Using fully extractability we can from c compute $im \bmod \mathrm{ord}(g)$ and from K we can compute $i \bmod \mathrm{ord}(g)$. Since i is invertible we can then compute $m \bmod \mathrm{ord}(g) = m$. For a 0/1-extractable generator the X-keys will be the elements of the form $K = g^i f(r_K)$, where $i \in \boldsymbol{Z}_{\mathrm{ord}(g)} \setminus \{0\}$. By the 0/1-extractability these keys can be efficiently sampled given t. For extraction, note that $\mathrm{commit}_K(0, r) \in F$ and $\mathrm{commit}_K(1, r) \notin F$ and use the 0/1-extractability of the generator. For the fully extractable construction and the 0/1-extractable construction, the indistinguishability of the key-spaces follows directly from the indistinguishability requirement on the generator. The transformed scheme is given by $\mathrm{commit}_{K_1,K_2}(m, (r_1, r_2, m_1)) = (K_1^{m_1} f(r_1), K_2^{m_2} f(r_2))$, where $m_2 = m - m_1 \bmod q$.

Proofs of Relations For the mixed commitment schemes we exhibit in this paper, there are efficient protocols for proving in zero-knowledge relations among committed values. As we shall see, it is possible to have the derived universally composable commitment schemes inherit these protocols while maintaining universal composability. In order for this to work, we need the protocols to be non-erasure Σ-protocols.

A non-erasure Σ-protocol for relation R is a protocol for two parties, called the prover P and the verifier V. The prover gets as input $(x, w) \in R$, the verifier gets as input x, and the goal is for the prover to convince the verifier that he knows w such that $(x, w) \in R$, without revealing information about w. We require that it is done using a protocol of the following form. The prover first computes a message $a \leftarrow A(x, w, r_a)$, where r_a is a uniformly random string, and sends a to V. Then V returns a random challenge e of length l. The prover then computes a responds to the challenge $z \leftarrow Z(x, w, r_a, e)$, and sends z to the verifier. The verifier then runs a program B on (x, a, e, z) which outputs $b \in \{0, 1\}$ indicating where to believe that the prover knows a valid witness w or not. Besides the protocol being of this form we furthermore require that the following hold:

Completeness If $(x, w) \in R$, then the verifier always accepts $(b = 1)$.

Special honest verifier zero-knowledge There exists a PPT algorithm, the honest verifier simulator hvs, which given instance x (where there exists w such that $(x, w) \in R$) and any challenge e generates $(a, z) \leftarrow \mathrm{hvs}(x, e, r)$, where r is a uniformly random string, such that (x, a, e, z) is distributed identically to a successful conversation where e occurs as challenge.

State construction Given (x, w, a, e, z, r), where $(a, z) = \text{hvs}(x, e, r)$ and $(x, w) \in R$ it should be possible to compute uniformly random r_a for which $a = A(x, w, r_a)$ and $z = Z(x, w, r_a, e)$.

Special soundness There exists a PPT algorithm, which given x, (a, e, z), and (a, e', z'), where $e \neq e'$, $B(x, a, e, z) = 1$, and $B(x, a, e', z') = 1$, outputs w such that $(x, w) \in R$.

In [**?**] it is shown how to use Σ-protocols in a concurrent setting. This is done by letting the first message be a commitment to a and then letting the third message be (a, r, z), where (a, r) is an opening of the commitment and z is computed as usual. If the commitment scheme used is a trapdoor commitment scheme this will allow for simulation using the honest verifier simulator. In an adaptive non-erasure setting, where an adversary can corrupt parties during the execution, it is also necessary with the State Construction property as the adversary is entitled to see the internal state of a corrupted party.

Proofs of Relations for the Schemes based on q One-Way Homomorphisms The basis for the proofs of relations between commitments will be the following proof of knowledge which works for fully extractable generators with public order, so we have $(G, H, f, g, q, b, b', t)$, with $b = \text{ord}(g) = q$. Assume that the prover is given $K \in H$, $m \in \boldsymbol{Z}_b$ and $r \in G$, and the verifier is given K and $C = K^m f(r)$. To prove knowledge of m, r, we do as follows:

1. The prover sends $\overline{C} = K^{\overline{m}} f(\overline{r})$ for uniformly random $\overline{m} \in \boldsymbol{Z}_q$ and $\overline{r} \in G$.
2. The verifier sends a uniformly random challenge e from $\boldsymbol{Z}_{b'}$, where b' is the public bound on the smallest primefactor in $\text{ord}(g)$.
3. The prover replies with $\tilde{m} = em + \overline{m} \bmod q$ and $\tilde{r} = f^{-1}(K^q)^{\tilde{i}} r^e \overline{r}$, where $\tilde{i} = em + \overline{m} \operatorname{div} q$. The verifier accepts iff $K^{\tilde{m}} f(\tilde{r}) = C^e \overline{C}$.

We argue that this is a non-erasure Σ-protocol: The completeness is immediate. For special soundness assume that we have two accepting conversations $(\overline{C}, e, \tilde{m}, \tilde{r})$ and $(\overline{C}, e', \tilde{m}', \tilde{r}')$. By the requirement that b' is smaller than the smallest primefactor of q we can compute α, β s.t. $1 = \alpha q + \beta(e - e')$. By our assumptions, we can compute $r_c = f^{-1}(C^q)$ and $r'_K = f^{-1}(K^q)$. Then compute $n = (\tilde{m} - \tilde{m}')\beta$, $m = n \bmod q$, and $r = (\tilde{r}/\tilde{r}')^\beta r_c^\alpha (r'_K)^{n \operatorname{div} q}$. Then $C = K^m f(r)$. For special honest verifier zero-knowledge, given C and e, pick $\tilde{m} \in \boldsymbol{Z}_q$ and $\tilde{r} \in G$ at random and let $\overline{C} = K^{\tilde{m}} f(\tilde{r}) C^{-e}$. For the state construction, assume that we are then given m, r such that $C = K^m f(r)$. Then let $\overline{m} = \tilde{m} - em \bmod q$, $\tilde{i} = em + \overline{m} \operatorname{div} q$, $\overline{r} = \tilde{r} f^{-1}(K^q)^{\tilde{i}} r^{-e}$. Then all values have the correct distribution.

We extend this scheme to prove relations between committed values. Assume that the prover knows $K_1, m_1, r_1, \ldots, K_l, m_l, r_l$ where $\sum_{i=1}^{l} a_i m_i = a_0 \bmod q$ for $a_0, \ldots, a_l \in \boldsymbol{Z}_q$, and assume that the verifier knows K_i and $C_i = K_i^{m_i} f(r_i)$ for $i = 1, \ldots, l$ and knows $a_0, \ldots, a_l$. The prover proves knowledge as follows: Run a proof of knowledge as that described above for each of the commitments using the same challenge e in them all. Let $\tilde{m}_i$ be the $\tilde{m}$-value of the protocol for C_i.

We furthermore instruct the verifier to check that $\sum_{i=1}^{l} a_i \tilde{m}_i = ea_0$. For special soundness assume that we have accepting conversations for the two challenges $e \neq e'$. Then we can compute $m_i = (\tilde{m}_i - \tilde{m}_i')(e-e')^{-1} \bmod q$ and r_i as above such that $C_i = K_i^{m_i} f(r_i)$. Furthermore $\sum_{i=1}^{l} a_i m_i = (e-e')^{-1}(\sum_{i=1}^{l} a_i \tilde{m}_i - \sum_{i=1}^{l} a_i \tilde{m}_i') = (e-e')^{-1}(ea_0 - e'a_0) = a_0$. The other properties of a non-erasure Σ-protocol follows using similar arguments.

This handles proofs of knowledge for the basic scheme. Recall, however, that in our UCC construction we need a transformed scheme where pairs of basic commitments are used, as described above. So assume, for instance, that we are given transformed commitments $(C_1, C_2), (C_3, C_4), (C_5, C_6)$ and we want to prove that the value committed to by (C_1, C_2) is the sum modulo q of the values committed by (C_3, C_4) and (C_5, C_6). This can be done by using the above protocol to prove knowledge of $m_1, \ldots, m_6$ contained in $C_1, \ldots, C_6$ such that $(m_1 + m_2) - (m_3 + m_4) - (m_5 + m_6) = 0$. All linear relations between transformed commitments can be dealt with in a similar manner.

By extending the proof of multiplicative relations from [**?**] in a manner equivalent to what we did for the additive proof we obtain a non-erasure Σ-protocol for proving multiplicative relations between transformed commitments.

Now for schemes without public order, the Σ-protocols given above do not directly apply because we were assuming that $b = \mathrm{ord}(g) = q$. However, we can modify the basic protocol by setting $m = \overline{m} = \tilde{m} = 0$. This results in a non-erasure Σ-protocol which allows the prover to prove knowledge of r, where $C = f(r)$ is known by the verifier. I.e. the prover can prove that $C \in F$, in other words that C commits to 0. Given $C = Kf(r)$ the prover can using the same protocol prove that $CK^{-1} \in F$, i.e. prove that $C \in KF$, in other words that C commits to 1. Using the technique from [**?**] for monotone logical combination of Σ-protocols we can then combine such proofs. Let $C_1, \ldots, C_l$ be commitments and let $R = \{(b_1^i, \ldots, b_l^i)\}_{i=1}^{a} \subset \{0,1\}^l$ be a Boolean relation. We can then prove that $C_1, \ldots, C_l$ commits to $(m_1, \ldots, m_l) \in R$ by proving $\bigvee_{i=1}^{a} \bigwedge_{i=1}^{l} C_i \in K^{m_i} F$. Let in particular $\mathbf{0} = \{(0,0),(1,1)\}$ and let $\mathbf{1} = \{(0,1),(1,0)\}$. Then proving knowledge of $(m_1, m_2) \in \mathbf{0}$ for transformed commitment $C = (C_1, C_2)$ proves that C commits to 0, similar for $\mathbf{1}$. Then using the relation $\mathbf{And} = \mathbf{0} \times \mathbf{0} \times \mathbf{0} \cup \mathbf{0} \times \mathbf{0} \times \mathbf{1} \cup \mathbf{0} \times \mathbf{1} \times \mathbf{0} \cup \mathbf{1} \times \mathbf{1} \times \mathbf{1}$, we can prove that three transformed commitments C_1, C_2, C_3 commits to bits m_1, m_2, m_3 s.t. $m_1 = m_2 \wedge m_3$. All Boolean relations of arity $O(\log(k))$ can handled in a similar manner. This will work for both fully and 0/1-extractable schemes.

The following theorem summarizes what we have argued:

Theorem 1. *If there exists a fully* (0/1) *extractable q one-way homomorphism generator, then there exists a special mixed commitment scheme with message space* $\boldsymbol{Z}_b$ *(*$\boldsymbol{Z}_2$*) as described above and with proofs of relations of the form* $m = f(m_1, m_2, \ldots, m_l)$ *where f is a Boolean predicate and* $l = O(\log(k))$*. If the scheme is with public order* $b = \mathrm{ord}(g) = q$ *and is fully extractable, we also have proofs of additive and multiplicative relations modulo q.*

3 Universally Composable Commitments

In the framework from [?] the security of a protocol is defined by comparing its real-life execution to an ideal evaluation of its desired behavior.

The protocol π is modeled by n interactive Turing Machines $P_1, \ldots, P_n$ called the parties of the protocol. In the real-life execution of π an adversary $\mathcal{A}$ and an environment $\mathcal{Z}$ modeling the environment in which $\mathcal{A}$ is attacking the protocol participates. The environment gives inputs to honest parties, receives outputs from honest parties, and can communication with $\mathcal{A}$ at arbitrary points in the execution. The adversary can see all messages and schedules all message deliveries. The adversary can corrupted parties adaptively. When a party is corrupted, the adversary learns the entire execution history of the corrupted party, including the random bits used, and will from the point of corruption send messages on behalf of the corrupted party. Both $\mathcal{A}$ and $\mathcal{Z}$ are PPT interactive Turing Machines.

Second an ideal evaluation is defined. In the ideal evaluation an ideal functionality $\mathcal{F}$ is present to which all the parties have a secure communication line. The ideal functionality is an interactive Turing Machine defining the desired input-output behavior of the protocol. Also present is an ideal model adversary $\mathcal{S}$, the environment $\mathcal{Z}$, and n so-called dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$ — all PPT interactive Turing Machines. The only job of the dummy parties is to take inputs from the environment and send them to the ideal functionality and take messages from the ideal functionality and output them to the environment. Again the adversary schedules all message deliveries, but can now not see the contents of the messages. This basically makes the ideal process a trivially secure protocol with the same input-output behavior as the ideal functionality. The framework also defines the hybrid models, where the execution proceeds as in the real-life execution, but where the parties in addition have access to an ideal functionality. An important property of the framework is that these ideal functionalities can securely be replaced with sub-protocols securely realizing the ideal functionality. The real-life model including access to an ideal functionality $\mathcal{F}$ is called the $\mathcal{F}$-hybrid model.

At the beginning of the protocol all parties, the adversary, and the environment is given as input the security parameter k and random bits. Furthermore the environment is given an auxiliary input z. At some point the environment stops activating parties and outputs some bit. This bit is taken to be the output of the execution. We use $\mathrm{REAL}_{\pi,\mathcal{A},\mathcal{Z}}(k,z)$ to denote the output of $\mathcal{Z}$ in the real-life execution and use $\mathrm{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)$ to denote the output of $\mathcal{Z}$ in the ideal evaluation. Let $\mathrm{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$ denote the distribution ensemble $\{\mathrm{REAL}_{\pi,\mathcal{A},\mathcal{Z}}(k,z)\}_{k\in\boldsymbol{N},z\in\{0,1\}^*}$ and let $\mathrm{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)$ denote the distribution ensemble $\{\mathrm{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)\}_{k\in\boldsymbol{N},z\in\{0,1\}^*}$.

Definition 2 ([?]). *We say that* π securely realizes $\mathcal{F}$ *if for all real-life adversaries* $\mathcal{A}$ *there exists an ideal model adversary* $\mathcal{S}$ *such that for all environments* $\mathcal{Z}$ *we have that* $\mathrm{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$ *and* $\mathrm{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$ *are computationally indistinguishable.*

An important fact about the above security notion is that it is maintained even if an unbounded number of copies of the protocol (and other protocols) are carried out concurrently — see [**?**] for a formal statement and proof. In proving the composition theorem it is used essentially that the environment and the adversary can communicate at any point in an execution. The price for this strong security notion, which is called universal composability in [**?**], is that rewinding cannot be used in the simulation.

The Commitment Functionality We now specify the task that we want to implement as an ideal functionality. We look at a slightly different version of the commitment functionality than the one in [**?**]. The functionality in [**?**] is only for committing to one bit. Here we generalize. The domain of our commitments will be the domain of the special mixed commitment used in the implementation. Therefore the ideal functionality must specify the domain by initially giving a system key N. For technical reasons, in addition, the X-trapdoor of N is revealed to the the ideal model adversary, i.e., the simulator. This is no problem in the ideal model since here the X-trapdoor cannot be used to find committed values – the ideal functionality stores committed values internally and reveals nothing before opening time. The simulator, however, needs the X-trapdoor in order to do the simulation of our implementation. The implementation, on the other hand, will of course keep the X-trapdoor of N hidden from the *real-life* adversary. The ideal functionality for homomorphic commitments is named $\mathcal{F}_{\mathrm{HCOM}}$ and is as follows.

0. Generate a uniformly random system key N along with the X-trapdoor t_N. Send N to all parties and send (N, t_N) to the adversary.
1. Upon receiving $(\text{commit}, sid, cid, P_i, P_j, m)$ from $\tilde{P}_i$, where m is in the domain of system key N, record (cid, P_i, P_j, m) and send the message $(\texttt{receipt}, sid, cid, P_i, P_j)$ to $\tilde{P}_j$ and the adversary. Ignore subsequent $(\text{commit}, sid, cid, \ldots)$ messages. The values sid and cid are a session id and a commitment id.
2. Upon receiving the message $(\texttt{prove}, sid, cid, P_i, P_j, R, cid_1, \ldots, cid_a)$ from $\tilde{P}_i$, where $(cid_1, P_i, P_j, m_1), \ldots, (cid_a, P_i, P_j, m_a)$ have been recorded, R is an a-ary relation with a non-erasure Σ-protocol, and $(m_1, m_2, \ldots, m_a) \in R$, send the message $(\texttt{prove}, sid, cid, P_i, P_j, R, cid_1, \ldots, cid_a)$ to $\tilde{P}_j$ and the adversary.
3. Upon receiving a message $(\texttt{open}, sid, cid, P_i, P_j)$ from $\tilde{P}_i$, where (cid, P_i, P_j, m) has been recorded, send the message $(\texttt{open}, sid, cid, P_i, P_j, m)$ to $\tilde{P}_j$ and the adversary.

It should be noted that a version of the functionality where N and t_N are not specified by the ideal functionality could be used. We could then let the domain of the commitments be a domain contained in (or easy to encode in) the domain of all the system keys.

The Common Reference String Model As mentioned in the introduction we cannot hope to construct two-party UCC in the plain real-life model. We need a that a common reference string (CRS) with a prescribed distribution is available to the players. It is straightforward to model a CRS as an ideal functionality $\mathcal{F}_{\mathrm{CRS}}$, see e.g. [**?**].

4 UCC with Constant Expansion Factor

Given a special mixed commitment scheme `com` we construct the following protocol $\mathrm{UCC}_{\texttt{com}}$.

The CRS The CRS is $(N, \overline{K}_1, \ldots, \overline{K}_n)$, where N is a random system key and $\overline{K}_1, \ldots, \overline{K}_n$ are n random E-keys for the system key N, $\overline{K}_i$ for P_i.

Committing

C.1 On input $(\mathrm{commit}, sid, cid, P_i, P_j, m)$ party P_i generates a random commitment key K_1 for system key N and commits to it as $c_1 = \mathrm{commit}_{\overline{K}_i}(K_1, r_1)$, and sends $(\texttt{com}_1, sid, cid, c_1)$ to P_j.[2]

R.1 P_j replies with $(\texttt{com}_2, sid, cid, K_2)$ for random key K_2.

C.2 P_i computes $K = K_1 + K_2$ and $c_2 = \mathrm{commit}_K(m, r_2)$ for random r_2, and records $(sid, cid, P_j, K, m, r_2)$ and sends the message $(\texttt{com}_3, sid, cid, K_1, r_1, c_2)$ to P_j.

R.2 P_j checks that $c_1 = \mathrm{commit}_{\overline{K}_i}(K_1, r_1)$, and if so computes $K = K_1 + K_2$, records (sid, cid, P_j, K, c_2), and outputs $(\texttt{receipt}, sid, cid, P_i, P_j)$.

Opening

C.3 On input $(\texttt{open}, sid, cid, P_i, P_j)$, P_i sends $(\texttt{open}, sid, cid, m, r_2)$ to P_j.

R.3 P_j checks that $c_2 = \mathrm{commit}_K(m, r_2)$, and if so outputs $(\texttt{open}, sid, cid, P_i, P_j, m)$.

Proving Relation

C.4 On input $(\texttt{prove}, sid, cid, P_i, P_j, R, cid_1, \ldots, cid_a)$, where $(sid, cid_1, P_j, K_1, m_1, r_1), \ldots, (sid, cid_a, P_j, K_a, m_a, r_a)$ are recorded commitments, compute the first message, a, of the Σ-protocol from the recorded witnesses and compute $c_3 = \mathrm{commit}_{\overline{K}_i}(a, r_3)$ for random r_3 and send $(\texttt{prv}_1, sid, cid, R, cid_1, \ldots, cid_a, c_3)$ to P_j.

R.4 P_j generates a random challenge e and sends $(\texttt{prv}_2, sid, cid, P_j, e)$ to P_i.

C.5 P_i computes the answer z and sends $(\texttt{prv}_3, sid, cid, a, r_3, z)$ to P_j.

R.5 P_j checks that $c_3 = \mathrm{commit}_{\overline{K}_i}(a, r_3)$ and that (a, e, z) is an accepting conversation. If so P_j outputs $(\texttt{prove}, sid, cid, P_i, P_j, R, cid_1, \ldots, cid_a)$.

Theorem 2. *If* `com` *is a special mixed commitment scheme, then the protocol* $\mathrm{UCC}_{\texttt{com}}$ *securely realizes* $\mathcal{F}_{\mathrm{HCOM}}$ *in the* CRS*-hybrid model.*

[2] We assume that the key space is a subset of the message space. If this is not the case the message space can be extended to a large enough $\mathcal{M}_N^l$ by committing to l values in the original scheme.

Proof. We construct a simulator $\mathcal{S}$ running a real-life adversary $\mathcal{A}$ and simulates to it a real-life execution consistent with the values input to and output from the ideal functionality in ideal-world in which $\mathcal{S}$ is running. The main requirements is that $\mathcal{S}$ given $|m|$ can simulate a commitment to m in such a way that it can later open the commitment to any value of m; That $\mathcal{S}$ can extract from the commitments given by $\mathcal{A}$ the value committed to; And that $\mathcal{S}$ does not rewind $\mathcal{A}$ as this is not possible in the model from [**?**].

The simulator $\mathcal{S}$ sets up the CRS s.t. the keys $\overline{K}_i$ are E-keys for which $\mathcal{S}$ knows the E-trapdoor, and such that the X-trapdoor is known too. When $\mathcal{S}$ is simulating a honest party acting as the committing party it use the E-trapdoor to open c_1 to $K_1 = K - K_2$, where K is generated as a random E-key with known E-trapdoor. Then $\mathcal{S}$ generates c_2 as an equivocable commitment, which it can later open to the actual committed value once it becomes known. When $\mathcal{S}$ is simulating a honest party acting as the receiver in a commitment it simply follows the protocol. Since no trapdoors are known to the adversary, the resulting key K will be random and in particular it will be an X-key except with negligible probability since all but a negligible fraction of the keys are X-keys. So, since $\mathcal{S}$ knows the X-trapdoor, it can compute from the c_2 sent by the adversary the value m committed to except with negligible probability.

For the proofs of relations, when $\mathcal{A}$ is giving a proof, $\mathcal{S}$ simply follows the protocol. The proofs given by honest P_i are simulated by $\mathcal{S}$. Here the non-rewinding simulation technique from [**?**] applies. If the party P_i is later corrupted, the messages which should have been committed to are learned. Using the E-trapdoor the simulator then opens the commitments in the relation appropriately. Then given the messages and the random bits (the witnesses of the proof), the state construction property of the proof allows to construct a consistent internal state to show to the adversary.

The main technical problem in proving this simulation indistinguishable from the real-life execution is that the X-trapdoor is used by the simulator, so we cannot do reductions to the computational binding of the mixed commitment scheme. We deal with this by defining a hybrid distribution that is generated by the following experiment: Run the simulation except do not use the X-trapdoor. Each time the adversary makes a commitment instead simply use the message 0 as input to the ideal functionality. When the adversary then opens the commitment to m, simply change the ideal-world communication to make it look as if m was the committed value. Up to the time of opening, the entire execution seen from the the environment and $\mathcal{A}$ is independent of whether 0 or m was given to the ideal functionality, and hence this hybrid is distributed exactly as the simulation. It is therefore enough to prove this hybrid indistinguishable from the real-life execution, which is possible using standard techniques. □

Perfect Hiding and Perfect Binding The scheme described above has neither perfect binding nor perfect hiding. Here we construct a version of the commitment scheme with both perfect hiding and perfect binding. The individual commitments are obviously not simultaneously perfect hiding and perfect binding, but it can be chosen at the time of commitment whether a commitment should be perfect binding or perfect hiding and proofs of relations can include both types of commitments. We sketch the scheme and the proof of its security. The details are left to the reader.

In the extended scheme we add to the CRS a random E-key K_E and a random X-key K_X (both for system key N). Then to do a perfect binding commitment to m the committer will in Step C.2 compute $c_2 = \mathrm{commit}_K(m, r_2)$ as before, but will in addition compute $c_3 = \mathrm{commit}_{K_X}(m, r_3)$. To open the commitment the committer will then have to send both a correct opening (m, r_2) of c_2 and a correct opening (m, r_3) of c_3. This is perfect binding as the X-key commitment is perfect binding.

To do a perfect hiding commitment the committer computes a uniformly random message $\overline{m}$ and commits with $c_2 = \mathrm{commit}_K(\overline{m} + m, r_2)$ and $c_3 = \mathrm{commit}_{K_E}(\overline{m}, r_3)$. To open to m the committer must then send a correct opening (m_2, r_2) of c_2 and a correct opening (m_3, r_3) of c_3 for which $m_2 = m_3 + m$. This is perfect hiding because c_3 hides $\overline{m}$ perfectly and $\overline{m} + m$ thus hides m perfectly.

To do the simulation simply let the simulator make the excusable mistake of letting K_E be a random X-key and letting K_X be a random E-key. This mistake will allow to simulate and cannot be detected by the fact that E-keys and X-keys are indistinguishable. For perfect binding commitments both K and K_X will then be E-keys when the simulator does a commitment, which allows to fake. When the adversary does a commitment K will (except with negligible) be an X-key and the simulator can extract m from $\mathrm{commit}_K(m)$. For perfect hiding commitments both K and K_E will (except with negligible probability) be X-keys when the adversary does a commitment, which allows to extract. When the simulator commits, K will be an E-key, which allows to fake an opening by faking $\mathrm{commit}_K(m)$.

For perfect binding commitments the proofs of relations can be used directly for the modified commitments by doing the proof on the $\mathrm{commit}_K(m)$ values. For perfect hiding commitments there is no general transformation that will carry proofs of relations over to the modified system. If however there is a proof of additive relation, then one can publish $\mathrm{commit}_K(m)$ and prove that the sum of the values committed to by $\mathrm{commit}_K(m)$ and $\mathrm{commit}_{K_E}(\overline{m})$ is committed to by $\mathrm{commit}_K(\overline{m} + m)$, and then use the commitment $\mathrm{commit}_K(m)$ when doing the proofs of relations.

5 Efficient Universally Composable Zero-Knowledge Proofs

In [**?**] Canetti and Fischlin showed how universally composable commitments can be used to construct simple zero-knowledge (ZK) protocols which are universally composable. This is a strong security property, which implies concurrent and non-malleable ZK proof of knowledge.

The functionality $\mathcal{F}_{\mathrm{ZK}}^R$ for universally composable zero-knowledge (for binary relation R) is as follows.

1. Wait to receive a value $(\texttt{verifier}, id, P_i, P_j, x)$ from some party P_i. Once such a value is received, send $(\texttt{verifier}, id, P_i, P_j, x)$ to $\mathcal{S}$, and ignore all subsequent $(\texttt{verifier}, \ldots)$ values.
2. Upon receipt of a value $(\texttt{prover}, id, P_j, P_i, x', w)$ from P_j, let $v = 1$ if $x = x'$ and $R(x, w)$ holds, and $v = 0$ otherwise. Send (id, v) to P_i and $\mathcal{S}$, and halt.

Exploiting the Multi-Bit Commitment Property In [?] a protocol for Hamiltonian-Cycle (HC) is given and proven to securely realize $\mathcal{F}_{\mathrm{ZK}}^{\mathrm{HC}}$. The protocol is of a common cut-and-choose form. It proceeds in t rounds. In each round the prover commits to l bits $m \in \{0,1\}^l$. Then the verifier sends a bit b as challenge. If $b = 0$ then the prover opens all commitments and if $b = 1$ the prover opens some subset of the challenges. Say that the subset is given by $S \in \{0,1\}^l$, where $S_i = 1$ if commitment number i should be revealed. Then if $b = 0$ the prover should see m and if $b = 1$ the prover should see $(S, m \wedge S)$. The verifier has two predicates V_0 and V_1 for verifying the reply from the prover. If $b = 0$ it verifies that $V_0(m) = 1$ and if $b = 1$ it verifies that $V_1(S, m \wedge S) = 1$. The protocol is such that seeing m or $(S, m \wedge S)$ reveals no knowledge about the witness (Hamiltonian cycle), but if $V_0(m) = 1$ and $V_1(S, m \wedge S) = 1$, then one can compute a witness from m and S. The verifier accepts if it can verify the reply in each of the t rounds. Obviously S should be kept secret when $b = 0$ — otherwise m and S would reveal the witness. This makes it hard to use the multi-bit commitments to commit to the l bits in such a way that just the subset S can be opened later. However, in [?] Kilian, Micali, and Ostrovski presented a general technique for transforming a multi-bit commitment scheme into a multi-bit commitment scheme with the property that individual bits can be open independently. Unfortunately their technique adds one round of interaction. However, we do not need the full generality of their result. This allows us to modify the technique to avoid the extra round of interaction.

We commit by generating a uniformly random pad $m_1 \in \{0,1\}^l$ and committing to the four values S, m_1, $m_2 = m \oplus m_1$, and $m_3 = m_1 \wedge S$ individually using multi-bit commitments. The verifier then challenges uniformly with $b \in \{0,1,2\}$. If $b = 0$, then reveal m_1 and m_2 and verify that $V_0(m_1 \oplus m_2) = 1$. If $b = 1$ then reveal S, m_2, m_3 and verify that $V_1(S, m_2 \wedge S \oplus m_3) = 1$. Finally, if $b = 2$, then reveal S, m_1, and m_3 and verify that $m_3 = m_1 \wedge S$. This is still secure as at no time are S and m revealed at the same time. For the soundness, note that if $V_0(m_1 \oplus m_2) = 1$, $V_1(S, m_2 \wedge S \oplus m_3) = 1$, and $m_3 = m_1 \wedge S$, then for $m = m_1 \oplus m_2$ we have that $V_0(m) = 1$ and $V_1(S, m \wedge S) = 1$ and can thus compute a witness. If we increase the number of rounds by a factor $\log_{3/2}(2) < 1.71$ we will get cheating probability no larger than for t rounds with cheating probability $1/2$ in each round. The number of bits committed to in each round is $4l$ for a total of less than $6.84tl$ bits. However, now the bits can be committed to k bits at a time using the multi-bit commitment scheme. Therefore, if we implement the modified protocol using our commitment scheme, we get communication complexity $O((l+k)t)$. This follows because we can commit to $O(l)$ bits by sending $O(l+k)$ bits. This is an improvement by a factor $\theta(\frac{lk}{l+k}) = \theta(\min(l,k))$ over [?].

Exploiting Efficient Proofs of Relations We show how we can use the efficient proofs of relations on committed values to reduce the communication complexity and the round complexity in a different way. This can simply be done by the parties agreeing in a Boolean circuit for the relation. Then the prover commits to the witness and the evaluation of the circuit on the witness and instance bit by bit and proves for each gate in the circuit that the committed values are consistent with the gate. The commitment to the output gate is opened and the prover then takes the revealed value as its output.

This protocol will have no messages of its own. All interaction is done through the ideal commitment functionality $\mathcal{F}_{\mathrm{HCOM}}$. Let l be the size of the gate used. This protocol requires $O(l)$ commitments to single bits, each of which require $O(k)$ bits of communication. Then we need to do $O(l)$ proofs of relations, each of which require $O(k)$ bits of communication. This amounts to $O(lk)$ bits of communication, and is an improvement over the $O(lkt)$ bits when using the scheme of [**?**] by a factor $O(t)$.

References

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42th Annual Symposium on Foundations of Computer Science*. IEEE, 2001.

[CD98] Ronald Cramer and Ivan Damgaard. Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free. In Hugo Krawczyk, editor, *Advances in Cryptology - Crypto '98*, pages 424–441, Berlin, 1998. Springer-Verlag. Lecture Notes in Computer Science Volume 1462.

[CDS94] R. Cramer, I. B. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology - Crypto '94*, pages 174–187, Berlin, 1994. Springer-Verlag. Lecture Notes in Computer Science Volume 839.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In J. Kilian, editor, *Advances in Cryptology - Crypto 2001*, pages 19–40, Berlin, 2001. Springer-Verlag. Lecture Notes in Computer Science Volume 2139.

[Dam00] Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In Bart Preneel, editor, *Advances in Cryptology - EuroCrypt 2000*, pages 418–430, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1807.

[KMO89] Joe Kilian, Silvio Micali, and Rafail Ostrovsky. Minimum resource zero-knowledge proofs (extended abstract). In *30th Annual Symposium on Foundations of Computer Science*, pages 474–479, Research Triangle Park, North Carolina, 30 October–1 November 1989. IEEE.

[OU98] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In K. Nyberg, editor, *Advances in Cryptology - EuroCrypt '98*, pages 308–318, Berlin, 1998. Springer-Verlag. Lecture Notes in Computer Science Volume 1403.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residue classes. In Jacques Stern, editor, *Advances in Cryptology - EuroCrypt '99*, pages 223–238, Berlin, 1999. Springer-Verlag. Lecture Notes in Computer Science Volume 1592.