16. Commiments Schemes

The goal of this part is to study various **commitment schemes**.

A commitment scheme is an essential primitive in cryptography, since it allows the committer to put a value in a box, so that nobody has any idea about the actual value (*hiding* property), but the committer cannot open the commitment in two different ways (*binding* property). It is also possible to prove relations between committed values, without revealing any additional information about them, excepted the fact that they satisfy these relations (*zero-knowledge* proofs).

More formally, a non-interactive commitment scheme is defined by three algorithms:

- $\mathsf{Setup}(1^k)$ outputs the public parameters of the system for a given security parameter k;
- $\mathsf{Commit}(m, r)$ takes the message m to commit to with some random coins r as inputs, and outputs the commitment c and an opening value d;
- $\mathsf{Verify}(c, m, d)$ takes the commitment c, the message m and an opening value d, and outputs yes or no, whether the verification succeeds or not.

The commitment c is sent to the receiver at the commit time, and the opening value d is sent together with the message m at the opening time, to allow verification.

The hiding property says that the commitment c does not leak information about m (either perfect secrecy, or computational indistinguishability), while the binding property says that no adversary (either powerful or computationally bounded) can generate c, $m \neq m'$ and d, d' such that both $\mathsf{Verify}(c, m, d)$ and $\mathsf{Verify}(c, m', d')$ accept.

In the following, we start with two simple commitment schemes, with complementary properties. The last part combines them to get a more powerful commitment scheme.

16.1 Pedersen Commitment

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q, and two random generators $g, h \in \mathbb{G}$. The Pedersen commitment scheme allows to commit to scalar elements from $\mathbb{Z}_q$:

Commitment: to commit to a scalar $m \in \mathbb{Z}_q$, one chooses a random $r \xleftarrow{\$} \mathbb{Z}_q$, and sets $c \leftarrow g^m h^r$, while the opening value is set to r;

Opening: to open a commitment $c \in \mathbb{G}$, one reveals the pair (m, r). If $c = g^m h^r$, the receiver accepts the opening to m, otherwise it refuses.

Q-1. Show that this commitment scheme is *perfectly hiding*: even a powerful adversary cannot have any idea about the committed value.

Q-2. Show that this commitment scheme is *computationally binding*: unless one can break a problem (to be specified), not adversary can open a commitment in two different ways.

Q-3. Show that this commitment scheme is *equivocal*: using a trapdoor (known to the simulator only, at the time of generation of the parameters (g, h), with an alternative but indistinguishable Setup algorithm), the simulator can generate a commitment $c \in \mathbb{G}$ so that it can open it later in any way of its choice.

Q-4. Under which condition can we use the binding property and the equivocality in a security proof?

Q-5. Under which condition can we use the hiding property and the equivocality in a security proof?

16.2 ElGamal Commitment

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q, and two random generators $g, h \in \mathbb{G}$. The ElGamal commitment scheme allows to commit to group elements from $\mathbb{G}$:

Commitment: to commit to a group element $M \in \mathbb{G}$, one chooses a random $r \xleftarrow{\$} \mathbb{Z}_q$, and sets $c \leftarrow (c_0 = g^r, c_1 = Mh^r)$, while the opening value is set to r;

Opening: to open a commitment $c \in \mathbb{G}$, one reveals the pair (M, r). If $c = (g^r, Mh^r)$, the receiver accepts the opening to M, otherwise it refuses.

Q-6. Show that this commitment scheme is *perfecly binding*: even a powerful adversary cannot open a commitment in two different ways.

Q-7. Show that this commitment scheme is *computationally hiding*: unless one can break a problem (to be specified), no adversary can distinguish commitments to M_0 or M_1 of its choice.

Q-8. Show that this commitment scheme is *extractable*: using a trapdoor ((known to the simulator only, at the time of generation of the parameters (g, h), with an alternative but indistinguishable Setup algorithm), the simulator can extract the committed value in any $c \in \mathbb{G}$.

Q-9. Under which condition can we use the binding property and the extractability in a security proof?

Q-10. Under which condition can we use the hiding property and the extractability in a security proof?

Q-11. This commitment scheme is called "ElGamal" Commitment, since this is the ElGamal encryption. How one could make the hiding property and the extractability compatible without any limitation?

16.3 Non-Interactive Commitments

Q-12. Show that a non-interactive commitment cannot be both *perfectly hiding* and *perfectly binding* (which would mean both hiding and binding against powerful adversaries).

An efficient construction in the random oracle model is $c = \mathsf{Commit}(m, r) = H(m, r)$, for a hash function $H : \{0,1\}^* \to \{0,1\}^{2k}$, on the message $m \in \{0,1\}^*$ to commit, with random coins $r \in \{0,1\}^{3k}$, for a security parameter k, while the opening value is set to r.

Q-13. Show that this is indeed a non-interactive commitment scheme (hiding and binding), and say under which assumptions (some limit or not on the number of queries to the random oracle).

Q-14. Show that it is also extractable and equivocal for a simulator that can access the list of query-answer pairs and that can program the random oracle in an indistinguishable way.

In order to build such an equivocal and extractable commitment scheme, in the standard model (without random oracles), let us start with two commitment schemes:

- an equivocal bit-commitment scheme $\mathsf{Commit}_{\mathsf{eq}}(b, r)$, for a bit $b \in \{0,1\}$ and random coins r, that outputs a commitment c, and the opening value $d \in \{0,1\}^{2k}$;

- an extractable commitment scheme $\mathsf{Commit}_{\mathsf{ext}}(D, r')$, for a bitstring $D \in \{0,1\}^{2k}$ and random coins r', that outputs a commitment c', and the opening value O.

We stress that the unique condition between the two commitment schemes is that the opening values of the former one can be committed with the latter (both in the same space $\{0,1\}^{2k}$).
On a message $m = (m_1, \ldots, m_\ell) \in \{0,1\}^\ell$ the commitment algorithm $\mathsf{Commit}(m, ((r_i)_i, (D'_i)_i, (r'_{i,b})_{i,b}))$ works as follows:

- for random coins r_i for $i = 1, \ldots, \ell$, set $(c_i, D_i) \leftarrow \mathsf{Commit}_{\mathsf{eq}}(m_i, r_i)$;
- for random coins $D'_i \xleftarrow{\$} \{0,1\}^{2k}$, set $d_{i,m_i} \leftarrow D_i$ and $d_{i,1-m_i} \leftarrow D'_i$, for $i = 1, \ldots, \ell$;
- for random coins $r'_{i,b}$, set $(c'_{i,b}, O_{i,b}) \leftarrow \mathsf{Commit}_{\mathsf{ext}}(d_{i,b}, r'_{i,b})$, for $i = 1, \ldots, \ell$ and $b = 0, 1$;
- output the commitment $(c_i, (c'_{i,b})_b)_i$, while the opening value is $(d_{i,m_i}, O_{i,m_i})_i$.

Q-15. Explain how works the Verify algorithm.

Q-16. Show this is indeed a commitment scheme: with both hiding and binding properties.

Q-17. Show this commitment scheme is also both equivocal and extractable.