1 Constructing Commitment Schemes

We now turn our attention to the construction of the first cryptographic primitive that we mentioned in the context of coin flipping.

1.1 Syntax of a commitment scheme

Let λ be the security parameter; we can think of this value as the key length. Abstractly the a commitment scheme can be divided in two stages: The commit stage and the open stage.

Commit Stage
1. Bob generates the parameters of the scheme and sends b to Alice
2. Alice selects her message M and commits to it by calculating $c \leftarrow \mathsf{Commit}(b, M)$
3. Alice sends c to Bob

Open Stage
4. Alice sends the revelation r and the message M to Bob
5. Bob runs the verification algorithm to ensure the opening of Alice is appropriate.

$$\mathsf{Verify}(r, M, c) = \begin{cases} 1 & accept \\ 0 & reject \end{cases}$$

1.2 Properties of the solution

Correctness: For every message M we require that: If $b \leftarrow Param(\lambda)$ and $(r, c) \leftarrow \mathsf{Commit}(b, M)$, then $\mathsf{Verify}(r, M, c) = 1$, where b, c, r are chosen randomly.

Binding: Intuitively, this property requires that Alice should not be in position to change her message after sending her commitment. More formally let A be an algorithm that can commit two different messages M_1, M_2 to the same commitment c, that is

Algorithm 1 $bindattack^A(1^\lambda)$

```
Let b ← GGen(1^λ)
(c, r_1, M_1, r_2, M_2) ← A(b)
if Verify(c, r_1, M_1) = 1 and Verify(c, r_2, M_2) = 1 and M_1 ≠ M_2 then
        return 1
else
        return 0
end if
```

We require that for every PPT A the following holds:

$$\mathsf{Prob}[bindattack^A(1^\lambda) = 1] = \mathsf{negl}(1^\lambda)$$

Hiding: The secrecy of Alice's message is preserved, meaning that Bob cannot extract any information about Alice's message

Algorithm 2 $hidingattack^{B}(1^{\lambda})$

1: Let $(a, M_1, M_2) \leftarrow B_1(1^{\lambda})$
2: $d \xleftarrow{R} \{0,1\}$
3: $(c, r) \leftarrow \mathsf{Commit}(b, M_d)$
4: $d^* \leftarrow B_2(c, a)$
5: **if** $d^* = d$ and $M_1 \neq M_2$ **then**
6: **return** 1
7: **else**
8: **return** 0
9: **end if**

We require that for every PPT $B = (B_1, B_2)$

$$\mathsf{Prob}[hidingattack^{B}(1^{\lambda}) = 1] \leq \frac{1}{2} + \mathsf{negl}(1^{\lambda})$$

1.3 Implementation

Now that we have specified what a commitment scheme is, it is time to see a way to implement it. We will describe Pedersen's protocol using the discrete logarithm assumption and prove its security. Before presenting the protocol's description, we will first define the Discrete Logarithm Problem. Suppose we have the group sampler that produces the description of a group G and of a cyclic subgroup of order m within G called GGen. In this section we will consider only modular groups over a prime number p; the description of G can be just the number p. We further consider only the case that m is a prime number. Based on this we define the following algorithm that samples discrete-logarithm parameters.

1. $\langle p, m, g\rangle \leftarrow \mathsf{GGen}(1^{\lambda})$
2. $t \xleftarrow{R} \mathbb{Z}_m$
3. $h \leftarrow g^t$

For an algorithm A, we say that it solves the Discrete Log Problem, if

$$\mathsf{Prob}[A(p, m, g, h) = t]$$

is not negligible. Based on the above we can now define the DLog assumption, according to which:

$$\forall \text{ PPT } A \; \mathsf{Prob}[A(p, m, g, h) = t] = \mathsf{negl}(\lambda)$$

We now describe Pedersen's protocol.

1. Bob runs $\mathsf{GGen}(1^{\lambda})$, outputs (p, m, g) such that p is prime, $g \in \mathbb{Z}_p^*$, $m = order(g)$ (that is $g^m = 1$), m is prime $|m| = \lambda$ bits, $m|p-1$ and $h = g^t$, where $t \xleftarrow{R} \mathbb{Z}_m$.
2. Bob sends (m, g, p, h) to Alice
3. Alice checks that p and m are primes, that $m \mid p-1$ and that $g^m = h^m = 1$. She selects $r \xleftarrow{R} \mathbb{Z}_m$ and her message $M \in \mathbb{Z}_m$. She commits to her message:
$$c = g^r h^M$$
and sends her commitment
4. At the Opening/Verification stage, Alice sends the revelation r and her message M and Bob verifies using the condition:
$$\text{if } c = g^r h^M \text{ then 1 else 0}$$

1.3.1 Proof of security

The aforementioned protocol is secure, i.e. the Binding and Hiding properties hold.

Suppose that the Binding property does not hold. Then there exists an algorithm A, that is successful in the binding attack described earlier. We will use this adversary, to construct an algorithm that breaks the discrete-logarithm assumption and therefore end up in a contradiction.

The fact that A can perform a successful binding attack, means that it can find two pairs (r_1, m_1) and (r_2, m_2) such that

$$\left.\begin{array}{rl} c = & g^{r_1}h^{m_1} \\ c = & g^{r_2}h^{m_2} \end{array}\right\} \Rightarrow g^{r_1}h^{m_1} = g^{r_2}h^{m_2}$$

which means that

$$g^{r_1-r_2} = h^{m_2-m_1}$$

Using the extended Euclidean Algorithm it is easy to find $k \in \mathbb{Z}_m$ such that $k(m_2 - m_1) = 1$, which means that

$$g^{(r_1-r_2)}k = h \mod m$$

and

$$(r_1 - r_2)k = \log_g h$$

But this contradicts our initial DLog assumption and therefore we can conclude, that such an Adversary does not exist and our protocol preserves the binding property.

We will now show that the hiding property holds. For this let us define a function

$$f : \mathbb{Z}_m \times \mathbb{Z}_m \to \langle g \rangle \equiv \mathbb{Z}_m$$

such that

$$f(r, M) = c$$

The above defined function f is surjective and bijective [1]

for every $M \in \mathbb{Z}_m$ [2]

It suffices to show that for $r_1, r_2 \xleftarrow{R} \mathbb{Z}_m$ and two messages M_1, M_2 the following holds:

$$\Delta[f(r_1, M_1), f(r_2, M_2)] = 0$$

or equivalently that

$$\Delta[f(r, M), U] = 0,$$

where U is the uniform distribution on $\mathbb{Z}_m$, which would evidently mean that c does not provide any information about M

Let $z \in \langle g \rangle$, i.e. $z = g^l$, where $l \in \mathbb{Z}_m$. Then we have that:

$$\begin{array}{rcl} \mathsf{Prob}[g(r, M) = z] & = & \mathsf{Prob}[g^r h^M = g^l] \\ & = & \mathsf{Prob}[g^r = g^l h^{-M} \\ & = & \mathsf{Prob}[g^r = g^{l-M}] \\ & = & \mathsf{Prob}[r = l - M] \\ & = & \dfrac{1}{m} \end{array}$$

[1] onto: Observe that $f(r, M) = g^r h^M = g^r (g^t)^M = g^{r+tM}$. So for $a \in \mathbb{Z}_m$ we have that there exists an r, such that $f(r, M) = a$

1-1: Suppose there exist r_1 and r_2 such that $f(r_1, M) = f(r_2, M)$ Then we have that $f(r_1, M) = f(r_2, M) \Rightarrow g^{r_1+T} = g^{r_2+T}$ for a $T = tM$

$\Rightarrow r_1 + T \equiv r_2 + T \mod m \Rightarrow r_1 \equiv r_2 \mod m$

Observe that we have used all the conditions, that Alice checks when she receives the parameters from Bob.

[2] Observe that if we allow M to vary, bijectivity breaks (take for example $(5, 0)$ and $(2, 1)$ in $\mathbb{Z}_7^*$ with $g = 3$)

Which proves that

$$\Delta[f(r, M), U] = 0,$$

We note that the security of the protocol is sensitive to the selection of parameters or to put it differently it is essential for Alice to check the parameters. Bob is the one who runs the $\mathsf{GGen}(1^\lambda)$ algorithm. If he chooses h such that $order(h) = 2m$ (for example $h = \zeta g^t$ and ζ such that $order(\zeta) = 2$ in $\mathbb{Z}_p$), then by receiving Alice's commitment c, he just needs to do the following in order to compute a single bit of M :

$$c^m = (g^r)^m (h^M)^m = (h^m)^M = \zeta^M = \begin{cases} 1, & M \text{ even} \\ \zeta, & M \text{ odd} \end{cases}$$

The above show that the test $h^m = 1$ is essential for the hiding property. What about the other tests that Alice performs ? can you find attacks if they are omitted ?

Notes by N. Karvelas