CHAPTER 20

Commitments and Oblivious Transfer

Chapter Goals

- To present two protocols which are carried out between mutually untrusting parties.
- To introduce commitment schemes and give simple examples of efficient implementations.
- To introduce oblivious transfer, and again give simple examples of how this can be performed in practice.

20.1. Introduction

In this chapter we shall examine a number of more advanced cryptographic protocols which enable higher-level services to be created. We shall particularly focus on protocols for

- commitment schemes,
- oblivious transfer.

Whilst there is a large body of literature on these protocols, we shall keep our feet on the ground and focus on protocols which can be used in real life to achieve practical higher-level services. It turns out that these two primitives are in some sense the most basic atomic cryptographic primitives which one can construct.

Up until now we have looked at cryptographic schemes and protocols in which the protocol participants are honest, and we are trying to protect their interests against an external adversary. However, in the real world we often need to interact with people who we do not necessarily trust. In this chapter we examine two types of protocol which are executed between two parties, each of whom may want to cheat in some way. The simplistic protocols in this chapter will form the building blocks on which more complicated protocols will be built in Chapters 21 and 22. We start by focusing on commitment schemes, and then we pass to oblivious transfer.

20.2. Commitment Schemes

Suppose Alice wishes to play "paper-scissors-stone" over the telephone with Bob. The idea of this game is that Alice and Bob simultaneously choose one of the set {`paper`,`scissors`,`stone`}. Then the outcome of the game is determined by the rules:

- `Paper` wraps `stone`. Hence if Alice chooses `paper` and Bob chooses `stone` then Alice wins.
- `Stone` blunts `scissors`. Hence if Alice chooses `stone` and Bob chooses `scissors` then Alice wins.
- `Scissors` cut `paper`. Hence if Alice chooses `scissors` and Bob chooses `paper` then Alice wins.

If both Alice and Bob choose the same item then the game is declared a draw. When conducted over the telephone we have the problem that whoever goes first is going to lose the game.

One way around this is for the party who goes first to "commit" to their choice, in such a way that the other party cannot determine what was committed to. Then the two parties can reveal their choices, with the idea that the other party can then verify that the revealing party has

N.P. Smart, *Cryptography Made Simple*, Information Security and Cryptography, DOI 10.1007/978-3-319-21936-3_20

not altered its choice between the commitment and the revealing stage. Such a system is called a *commitment scheme*. An easy way to do this is to use a cryptographic hash function as follows:

$$\begin{aligned} A &\longrightarrow B : h_A = H(R_A \| \texttt{paper}), \\ B &\longrightarrow A : \texttt{scissors}, \\ A &\longrightarrow B : R_A, \texttt{paper}. \end{aligned}$$

At the end of the protocol Bob needs to verify that the h_A sent by Alice is equal to $H(R_A\|\texttt{paper})$. If the values agree he knows that Alice has not cheated. The result of this protocol is that Alice loses the game since `scissors` cut `paper`.

Let us look at the above from Alice's perspective. She first commits to the value `paper` by sending Bob the hash value h_A. This means that Bob will not be able to determine that Alice has committed to the value `paper`, since Bob does not know the random value of R_A used and Bob is unable to invert the hash function. The fact that Bob cannot determine what value was committed to is called the concealing or hiding property of a commitment scheme.

As soon as Bob sends the value `scissors` to Alice, she knows she has lost but is unable to cheat, since to cheat she would need to come up with a different value of R_A, say R'_A, which satisfied

$$H(R_A\|\texttt{paper}) = H(R'_A\|\texttt{stone}).$$

But this would mean that Alice could find collisions in the hash function, which for a suitably chosen hash function is believed to be impossible. Actually we require that the hash function is second preimage resistant in this case. This property of the commitment scheme, that Alice cannot change her mind after the commitment procedure, is called binding.

Let us now study these properties of concealing and binding in more detail. Recall that an encryption function has information-theoretic security if an adversary with infinite computing power could not break the scheme, whilst an encryption function is called computationally secure if it is only secure when faced with an adversary with polynomially bounded computing power. A similar division can be made with commitment schemes, but now we have two security properties, namely concealing and binding. One property protects the interests of the sender, and one property protects the interests of the receiver. To simplify our exposition we shall denote our abstract commitment scheme by a public algorithm, $c = C(x, r)$ which takes a value $x \in \mathbb{P}$ and some randomness $r \in \mathbb{R}$ and produces a commitment $c \in \mathbb{C}$. To decomitment the commiter simply reveals the values of x and r. The receiver then checks that the two values produce the original commitment.

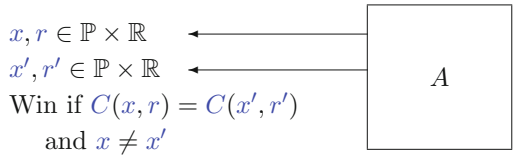

FIGURE 20.1. Commitment scheme: binding game

Definition 20.1 (Binding)**.** *A commitment scheme is said to be information-theoretically (resp. computationally)* binding *if no infinitely powerful (resp. computationally bounded) adversary can win the following game.*

- *The adversary outputs values $x \in \mathbb{P}$ and $r \in \mathbb{R}$.*
- *The adversary must then output a value $x' \neq x$ and a value $r' \in \mathbb{R}$ such that*

$$C(x, r) = C(x', r').$$

This game is given graphically in Figure 20.1. If the commitment scheme Π is *computationally* binding then we can define an advantage statement which is defined as

$$\text{Adv}_{\Pi}^{\mathsf{bind}} = \Pr[A \text{ wins the binding game}].$$

For information-theoretically binding schemes such an advantage will be zero by definition.

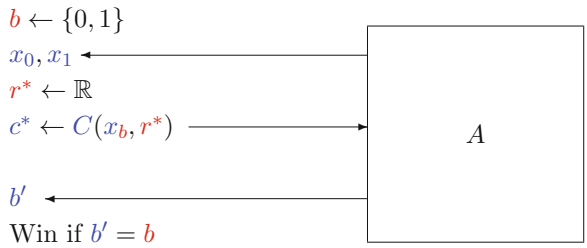

FIGURE 20.2. Commitment scheme: concealing game

Definition 20.2 (Concealing)**.** *A commitment scheme is said to be information-theoretically (resp. computationally)* concealing *if no infinitely powerful (resp. computationally bounded) adversary can win the following game.*

- *The adversary outputs two messages x_0 and x_1 of equal length.*
- *The challenger generates $r^* \in \mathbb{R}$ at random and a random bit $b \in \{0,1\}$.*
- *The challenger computes $c^* = C(x_b, r^*)$ and passes c^* to the adversary.*
- *The adversary's goal now is to guess the bit b.*

This game is defined in Figure 20.2. Just as with the binding property, if the commitment scheme Π is *computationally* concealing then we can define an advantage statement which is defined as

$$\text{Adv}_{\Pi}^{\mathsf{conceal}} = 2 \cdot \left| \Pr[A \text{ wins the concealing game}] - \frac{1}{2} \right|.$$

For information-theoretically concealing schemes such an advantage will be zero by definition. Notice that this definition of concealing is virtually identical to our definition of indistinguishability of encryptions. A number of results trivially follow from these two definitions.

Lemma 20.3. *There exists no scheme which is both information-theoretically concealing and binding.*

PROOF. Suppose we have a scheme which is both information-theoretically concealing and binding, and suppose the committer makes a commitment $c \leftarrow C(x, r)$. Since it is information-theoretically concealing there must exist values x' and r' such that $c = C(x', r')$; otherwise an infinitely powerful receiver could break the concealing property. But if Alice is also infinitely powerful this means she can break the binding property as well. □

Lemma 20.4. *Using the commitment scheme defined as*

$$c \leftarrow H(r \| m),$$

for a random value r, the committed value m and some cryptographic hash function H is at best

- *computationally binding,*
- *information-theoretically concealing.*

PROOF. All cryptographic hash functions we have met are only computationally secure against preimage resistance and second preimage resistance. The binding property of the above scheme

is only guaranteed by the second preimage resistance of the underlying hash function. Hence, the binding property is only computationally secure.

The concealing property of the above scheme is only guaranteed by the preimage resistance of the underlying hash function. Hence, the concealing property looks like it should be only computationally secure. However, if we assume that the value r is chosen from a suitably large set, then the fact that the hash function should have many collisions works in our favour and in practice we should obtain something "close" to information-theoretic concealing. On the other hand if we assume that H is a random oracle, then the commitment scheme is clearly information-theoretically concealing. □

We now turn to three practical commitment schemes which occur in various real-world protocols. All are based on a finite abelian group G of prime order q, which is generated by g. Two of the schemes will also require another generator $h \in \langle g \rangle$, where the discrete logarithm of h to the base g is unknown by any user in the system. To generate g and h we need to ensure that no one knows the discrete logarithm, and hence it needs to be done in a verifiably random manner.

Verifiably random generation of g and h is quite easy to ensure, for example for a finite field $\mathbb{F}_p^*$ with q dividing $p-1$ we create g as follows (with a similar procedure being used to determine h):

- $r \leftarrow \mathbb{Z}$.
- $f \leftarrow H(r) \in \mathbb{F}_p^*$ for some cryptographic hash function H.
- $g \leftarrow f^{(p-1)/q} \pmod{p}$.
- If $g = 1$ then return to the first stage, else output (r, g).

This generates a random element of the subgroup of $\mathbb{F}_p^*$ of order q, with the property that it is generated verifiably at random since one outputs the seed r used to generate the random element. Thus anyone who wishes to verify that g was generated in the above manner can use r to rerun the algorithm. Since we have used a cryptographic hash function H we do not believe that it is feasible for anyone to construct an r which produces a group element whose discrete logarithm is known with respect to some other group element.

Given g, h we define two commitment schemes, $B(x)$ and $B_a(x)$, to commit to an integer x modulo q, and one, $E_a(x)$, to commit to an element $x \in \langle g \rangle$.

$$\begin{aligned} B(x) &= g^x, \\ E_a(x) &= (g^a, x \cdot h^a), \\ B_a(x) &= h^x \cdot g^a, \end{aligned}$$

where a is a random integer modulo q. To reveal the commitments the user publishes the value x in the first scheme and the pair (a, x) in the second and third schemes. The value a is called the blinding value, since it blinds the value of the commitment x even to a computationally unbounded adversary. The scheme given by $B_a(x)$ is called Pedersen's commitment scheme.

Lemma 20.5. *The commitment scheme $B(x)$ is information-theoretically binding.*

Proof. Suppose Alice having published $c = B(x) = g^x$ wishes to change her mind as to which element of $\mathbb{Z}/q\mathbb{Z}$ she wants to commit to. Alas, for Alice no matter how much computing power she has there is mathematically only one element in $\mathbb{Z}/q\mathbb{Z}$, namely x, which is the discrete logarithm of the commitment c to the base g. Hence, the scheme is clearly information-theoretically binding. □

Note that this commitment scheme does not meet our strong definition of security for the concealing property, in any way; after all it is deterministic. If the space of values from which x is selected

is large, then this commitment scheme could meet a weaker security definition related to a one-way-like property. We leave the reader to produce a suitable definition for a one-way concealing property, and show that $B(x)$ meets this definition.

Lemma 20.6. *The commitment scheme $E_a(x)$ is information-theoretically binding and computationally concealing.*

PROOF. This scheme is exactly ElGamal encryption with respect to a public key h, for which *no one* knows the associated private key. Indeed any IND-CPA secure public key encryption scheme can be used in this way as a commitment scheme.

The underlying IND-CPA security implies that the resulting commitment scheme is computationally concealing, whilst the fact that the decryption is unique implies that the commitment scheme is information-theoretically binding. □

Lemma 20.7. *The Pedersen commitment scheme, given by $B_a(x)$, is computationally binding and information-theoretically concealing.*

PROOF. Suppose the adversary, after having committed to $c \leftarrow B_a(x) = h^x \cdot g^a$ wishes to change her mind, so as to commit to y instead. So the adversary outputs another pair (y, b) such that $c = h^y \cdot g^b$. However, given these two values we can extract the discrete logarithm of h with respect to g via

$$\frac{a-b}{y-x}.$$

Thus any algorithm which breaks the binding property can be turned into an algorithm which solves discrete logarithms in the group G.

We now turn to the concealing property. It is clear that this is information-theoretically concealing since an all-powerful adversary could extract the discrete logarithm of h with respect to g and then *any* committed value c can be opened to *any* message x. □

We end this section by noticing that the two discrete-logarithm-based commitment schemes we have given possess the homomorphic property:

$$\begin{aligned} B(x_1) \cdot B(x_2) &= g^{x_1} \cdot g^{x_2} \\ &= g^{x_1+x_2} \\ &= B(x_1 + x_2), \\ B_{a_1}(x_1) \cdot B_{a_2}(x_2) &= h^{x_1} \cdot g^{a_1} \cdot h^{x_2} \cdot g^{a_2} \\ &= h^{x_1+x_2} \cdot g^{a_1+a_2} \\ &= B_{a_1+a_2}(x_1 + x_2). \end{aligned}$$

We shall use this additively homomorphic property when we discuss an electronic voting protocol at the end of Chapter 21.

20.3. Oblivious Transfer

We now consider another type of basic protocol, called oblivious transfer or OT for short. This is another protocol which is run between two distrusting parties, a sender and a receiver. In its most basic form the sender has two secret messages as input, m_0 and m_1; the receiver has as input a single bit b. The goal of an OT protocol is that at the end of the protocol the sender should not learn the value of the receiver's input b. However, the receiver should learn the value of m_b but should learn nothing about m_{1-b}. Such a protocol is often called a 1-out-of-2 OT, since the receiver learns one of the two inputs of the sender. Such a protocol is depicted in Figure 20.3. One

can easily generalize this concept to a k-out-of-n OT, but as it is we will only be interested in the simpler case.

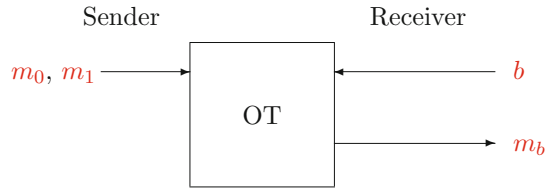

FIGURE 20.3. A 1-out-of-2 OT

We present a scheme which allows us to perform a 1-out-of-2 oblivious transfer of two arbitrary bit strings m_0, m_1 of equal length. The scheme is based on an IND-CPA version of DHIES, where we use simply the exclusive-or of the plaintext with a hash of the underlying Diffie–Hellman key to encryt the payload.

We take a standard discrete-logarithm-based public/private key pair $(h \leftarrow g^x, x)$, where g is a generator of cyclic finite abelian group G of prime order q. We will require a hash function H from G to bit strings of length n. Then to encrypt messages m of length n we compute, for a random $k \in (\mathbb{Z}/q\mathbb{Z})$,

$$c = (c_1, c_2) \leftarrow \left(g^k, m \oplus H(h^k)\right).$$

To decrypt we compute

$$c_2 \oplus H({c_1}^x) = m \oplus H(g^{kx}) = m \oplus H(h^k) = m.$$

It can easily be shown that the above scheme is semantically secure under chosen plaintext attacks (i.e. passive attacks) in the random oracle model.

The idea behind our oblivious transfer protocol is for the receiver to create two public keys h_0 and h_1, for only one of which does he know the corresponding secret key. If the receiver knows the secret key for h_b, where b is the bit he is choosing, then he can decrypt for messages encrypted under this key, but not decrypt under the other key. The sender then only needs to encrypt his messages with the two keys. Since the receiver only knows one secret key he can only decrypt one of the messages.

To implement this idea concretely, the sender first selects a random element c in G; it is important that the receiver does not know the discrete logarithm of c with respect to g. This value is then sent to the receiver. The receiver then generates two public keys, according to his bit b, by first generating $x \in (\mathbb{Z}/q\mathbb{Z})$ and then computing

$$h_b \leftarrow g^x, \quad h_{1-b} \leftarrow c/h_b.$$

Notice that the receiver knows the underlying secret key for h_b, but he does not know the secret key for h_{1-b} since he does not know the discrete logarithm of c with respect to g. These two public key values are then sent to the sender. The sender then encrypts message m_0 using the key h_0 and message m_1 using key h_1, i.e. the sender computes

$$c_0 \leftarrow \left(g^{k_0}, m_0 \oplus H({h_0}^{k_0})\right),$$
$$c_1 \leftarrow \left(g^{k_1}, m_1 \oplus H({h_1}^{k_1})\right),$$

for two random integers k_0, $k_1 \in (\mathbb{Z}/q\mathbb{Z})$. These two ciphertexts are then sent to the receiver who then decrypts the bth one using his secret key x.

From the above description we can obtain some simple optimizations. Firstly, the receiver does not need to send both h_0 and h_1 to the sender, since the sender can always compute h_1 from h_0 by

computing c/h_0. Secondly, we can use the same value of $k = k_0 = k_1$ in the two encryptions. We thus obtain the following oblivious transfer protocol:

Sender		Receiver
$c \leftarrow G$	$\xrightarrow{c}$	
		$x \leftarrow (\mathbb{Z}/q\mathbb{Z})$
		$h_b \leftarrow g^x$
	$\xleftarrow{h_0}$	$h_{1-b} \leftarrow c/h_b$
$h_1 \leftarrow c/h_0$		
$k \leftarrow (\mathbb{Z}/q\mathbb{Z})$		
$c_1 \leftarrow g^k$		
$e_0 \leftarrow m_0 \oplus H({h_0}^k)$		
$e_1 \leftarrow m_1 \oplus H({h_1}^k)$	$\xrightarrow{c_1,e_0,e_1}$	
		$m_b \leftarrow e_b \oplus H({c_1}^x)$.

So does this respect the two conflicting security requirements of the participants? First, note that the sender cannot determine the hidden bit b of the receiver since the value h_0 sent from the receiver is simply a random element in G. Then we note that the receiver can learn nothing about m_{1-b} since to do this they would have to be able to compute the output of H on the value h_{1-b}^k, which would imply contradicting the fact that H acts as a random oracle or being able to solve the Diffie–Hellman problem in the group G.

Chapter Summary

- We introduced the idea of protocols between mutually untrusting parties, and introduced commitment and oblivious transfer as two simple examples of such protocols.
- A commitment scheme allows one party to bind themselves to a value, and then reveal it later.
- A commitment scheme needs to be both binding and concealing. Efficient schemes exist which are either information-theoretically binding or information-theoretically concealing, but not both.
- An oblivious transfer protocol allows a sender to send one of two messages to a recipient, but she does not know which message is actually received. The receiver also learns nothing about the other message which was sent.

Further Reading

The above oblivious transfer protocol originally appeared in a slightly modified form in the paper by Bellare and Micali. The paper by Naor and Pinkas discusses a number of optimizations of the oblivious transfer protocol which we presented above. In particular it presents mechanisms to efficiently perform 1-out-of-N oblivious transfer. The papers by Blum and Shamir et al. provide some nice early ideas related to commitment schemes.

M. Bellare and S. Micali. *Non-interactive oblivious transfer and applications.* In Advances in Cryptology – Crypto 1989, LNCS 435, 547–557, Springer, 1990.

M. Blum. *Coin flipping by telephone.* SIGACT News, **15**, 23–27, 1983.

M. Naor and B. Pinkas. *Efficient oblivious transfer protocols.* In SIAM Symposium on Discrete Algorithms – SODA 2001, 448–457, SIAM, 2001.

A. Shamir, R. Rivest and L. Adleman. *Mental Poker.* The Mathematical Gardner, 37–43, Prindle, Weber and Schmidt, 1981.