Chapter 11

Asymmetric Encryption

The setting of public-key cryptography is also called the "asymmetric" setting due to the asymmetry in key information held by the parties. Namely one party has a secret key while another has the public key that matches this secret key. This is in contrast to the symmetry in the private key setting, where both parties had the same key. Asymmetric encryption is thus another name for public-key encryption, the mechanism for achieving data privacy in the public key or asymmetric setting.

Our study of asymmetric encryption (following our study of other primitives) will begin by searching for appropriate notions of security, and models and formalizations via which they are captured. We then consider constructions, where we look at how to design and analyze various schemes.

With regard to notions of security, we will be able to build considerably on our earlier study of symmetric encryption. Indeed, from this point of view there is very little difference between symmetric and asymmetric encryption; not much more than the fact that in the latter the adversary gets the public key as input. This is important (and re-assuring) to remember. All the intuition and examples we have studied before carry over, so that we enter the study of asymmetric encryption already having a good idea of what encryption is, how security is modeled, and what it means for a scheme to be secure. Accordingly we will deal with the security issues quite briefly, just re-formulating the definitions we have seen before.

The second issue (namely constructions) is a different story. Designs of asymmetric encryption schemes rely on tools and ideas different from those underlying the design of symmetric encryption schemes. Namely in the asymmetric case, the basis is (typically) computationally intractable problems in number theory, while for the symmetric case we used block ciphers. Thus, the greater part of the effort in this chapter will be on schemes and their security properties.

11.1 Asymmetric encryption schemes

An asymmetric encryption scheme is just like a symmetric encryption scheme except for an asymmetry in the key structure. The key pk used to encrypt is different from the key sk used to decrypt. Furthermore pk is public, known to the sender and also to the adversary. So while only a receiver in possession of the secret key can decrypt, anyone in possession of the corresponding public key can encrypt data to send to this one receiver.

Definition 11.1.1 An *asymmetric encryption scheme* $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ consists of three algorithms, as follows:

- The randomized *key generation* algorithm $\mathcal{K}$ (takes no inputs and) returns a pair (pk, sk) of keys, the public key and matching secret key, respectively. We write $(pk, sk) \xleftarrow{\$} \mathcal{K}$ for the operation of executing $\mathcal{K}$ and letting (pk, sk) be the pair of keys returned.
- The *encryption* algorithm $\mathcal{E}$ takes the public key pk and a *plaintext* (also called a *message*) M to return a value called the *ciphertext*. The algorithm may be randomized, but not stateful. We write $C \xleftarrow{\$} \mathcal{E}_{pk}(M)$ or $C \xleftarrow{\$} \mathcal{E}(pk, M)$ for the operation of running $\mathcal{E}$ on inputs pk, M and letting C be the ciphertext returned.
- The deterministic *decryption* algorithm $\mathcal{D}$ takes the secret key sk and a ciphertext $C \neq \perp$ to return a message M. We write $M \leftarrow \mathcal{D}_{sk}(C)$ or $M \leftarrow \mathcal{D}(sk, C)$.

The *message space* associated to a public key pk is the set $\mathsf{Plaintexts}(pk)$ of all M for which $\mathcal{E}_{pk}(M)$ never returns $\perp$. We require that the scheme provide *correct decryption*, which means that for any key-pair (pk, sk) that might be output by $\mathcal{K}$ and any message $M \in \mathsf{Plaintexts}(pk)$, if C was returned by $\mathcal{E}_{pk}(M)$ then $\mathcal{D}_{sk}(C) = M$. ▌

Let R be an entity that wants to be able to receive encrypted communications. The first step is key generation: R runs $\mathcal{K}$ to generate a pair of keys (pk, sk) for itself. Note the key generation algorithm is run locally by R. Anyone in possession of R's public key pk can then send a message M privately to R. To do this, they would encrypt M via $C \leftarrow \mathcal{E}_{pk}(M)$ and send the ciphertext C to R. The latter will be able to decrypt C using sk via $M \leftarrow \mathcal{D}_{sk}(C)$.

Note that an entity wishing to send data to R must be in possession of R's public key pk, and must be assured that the public key is authentic, meaning really is the R's public-key, and not someone else's public key. We will look later into mechanisms for assuring this state of knowledge. But the key management processes are not part of the asymmetric encryption scheme itself. In constructing and analyzing the security of asymmetric encryption schemes, we make the assumption that any prospective sender is in possession of an authentic copy of the public key of the receiver. This assumption is made in what follows.

A viable scheme of course requires some security properties. But these are not our concern now. First we want to pin down what constitutes a specification of a scheme, so that we know what are the kinds of objects whose security we want to assess.

The key usage is the "mirror-image" of the key usage in a digital signature scheme. In an asymmetric encryption scheme, the holder of the secret key is a receiver, using the secret key to decrypt ciphertexts sent to it by others. In a digital signature scheme, the holder of the secret key is a sender, using the secret key to tag its own messages so that the tags can be verified by others.

The last part of the definition says that ciphertexts that were correctly generated will decrypt correctly.

The encryption algorithm might be randomized, and must for security. But unlike in a symmetric encryption scheme, we will not consider stateful asymmetric encryption algorithms. This is because there is no unique sender to maintain state; many different entities are sending data to the receiver using the same public key. The decryption algorithm is deterministic and stateless.

We do not require that the message or ciphertext be strings. Many asymmetric encryption schemes are algebraic or number-theoretic, and in the natural formulation of these schemes messages might be group elements and ciphertexts might consist of several group elements. However, it is understood that either messages or ciphertexts can be encoded as strings wherever necessary. (The encodings will usually not be made explicit.) In particular, we might talk of the length of a

message of ciphertext, with the understanding that we mean the length of some binary encoding of the quantity in question. (We do this, for example, in defining security.)

In cases where messages are not strings, but, say, group elements, using the scheme in practice will usually require encoding of actual messages as group elements. We will discuss this as it arises.

11.2 Notions of security

Security of an encryption scheme (whether symmetric or asymmetric) is supposed to reflect the inability of an adversary, given ciphertexts (and any public information such as a public key), to get "non-trivial" information about the underlying plaintexts. We allow an adversary (having the goal of figuring out some non-trivial information about plaintexts from ciphertexts) different "attack" capabilities reflecting different situations. The most basic kind of attack is a chosen-plaintext attack, in which the adversary can obtain encryptions of messages of its choice. We discussed this type of attack in depth in the context of symmetric encryption, and argued that the definition of security in the sense of "left-or-right" captured security against these types of attacks in a strong sense. In the asymmetric case, the same is true, and we will use the same notion to capture security against chosen plaintext attack. (A difference that must be kept in mind is that the adversary in an asymmetric setting also has the public key, and so can in any case encrypt on its own, but this does not really affect the formalization of the notion.)

We also discussed the stronger chosen-ciphertext attack, in which we desire that privacy of data be maintained even if the adversary has some (limited) access to a "decryption oracle", this being a box that contains the secret decryption key and implements decryption under this key. (The adversary does not get the key itself.) For the asymmetric setting, chosen-ciphertext attacks are both more relevant and more difficult to protect against than in the symmetric setting.

We begin by summarizing the notion of security against chosen-plaintext attack, extending the definitions for the symmetric setting. Then we go on to discuss chosen-ciphertext attacks.

11.2.1 Security against chosen-plaintext attack

Let us fix a specific asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. We consider an adversary A that is an algorithm (program) that is given as input a public key pk. The intuition behind the notion is as follows. Imagine that the sender has two sequences of messages, $M_0^1, \ldots, M_0^q$ and $M_1^1, \ldots, M_1^q$. It encrypts the messages in one of the sequences to get a sequence of ciphertexts which it transmits. That is, if $b \in \{0,1\}$ denotes the choice of sequence, the sender computes $C^i \leftarrow \mathcal{E}_{pk}(M_b^i)$ for $i = 1, \ldots, q$, and then transmits $C^1, \ldots, C^q$ to the receiver. The adversary, being able to eavesdrop, obtains the ciphertexts. Its goal is to figure out which of the two message sequences was encrypted, namely to figure out the value of the bit b. The scheme is said to be "secure" if it cannot compute the value of b correctly with probability significantly more than $1/2$.

The formalization allows the adversary to specify both message sequences, and furthermore to mount an adaptive attack, meaning to choose M_0^i, M_0^{i-1} as a function of $C^1, \ldots, C^{i-1}$.

The formalization is in terms of the left-or-right encryption oracle. It depends on the public key and challenge bit b. It takes input two messages and returns a ciphertext, as follows:

Oracle $\mathcal{E}_{pk}(\mathbf{LR}(M_0, M_1, b))$ // $b \in \{0,1\}$ and $M_0, M_1 \in \{0,1\}^*$
  If $|M_0| \neq |M_1|$ then return $\perp$
  $C \leftarrow \mathcal{E}_K(M_b)$
  Return C

Thus the oracle encrypts one of the messages, the choice of which being made according to the bit b. Now we consider two "worlds":

World 0: The oracle provided to the adversary is $\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,0))$. So, whenever the adversary makes a query (M_0, M_1) to its oracle, the oracle computes $C \xleftarrow{\$} \mathcal{E}_{pk}(M_0)$, and returns C as the answer.

World 1: The oracle provided to the adversary is $\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,1))$. So, whenever the adversary makes a query (M_0, M_1) to its oracle, the oracle computes $C \xleftarrow{\$} \mathcal{E}_{pk}(M_1)$, and returns C as the answer.

We call the first world (or oracle) the "left" world (or oracle), and we call the second world (or oracle) the "right" world (or oracle). The problem for the adversary is, after talking to its oracle for some time, to tell which of the two oracles it was given.

The adversary queries makes some number of queries to its oracle, and then outputs a bit. This bit has some probability of equaling one. The probability is over the choice of the keys (pk, sk) as made by the key-generation algorithm, any random choices made by the oracle, and any other random choices made by the adversary in its computation. We look at this probability in each of the two worlds as the basis for the definition.

We suggest that the reader return to the chapter on symmetric encryption to refresh his or her mind about this model. In particular remember that the encryption function is randomized, and the oracle implementing it is thus randomized too. Each time the oracle computes a ciphertext, it does so by running the encryption algorithm with fresh coins.

Definition 11.2.1 Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an asymmetric encryption scheme, let $b \in \{0,1\}$, and let A be an algorithm that has access to an oracle and returns a bit. We consider the following experiment:

$$\begin{array}{l}\text{Experiment } \mathbf{Exp}^{\text{ind-cpa-}b}_{\mathcal{AE}}(A)\\ \quad (pk, sk) \xleftarrow{\$} \mathcal{K}\\ \quad b' \leftarrow A^{\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,b))}(pk)\\ \quad \text{Return } b'\end{array}$$

The *ind-cpa-advantage* of A is defined as

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A) \;=\; \Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE}}(A) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE}}(A) = 1\right] \;. \quad \blacksquare$$

As usual, the time-complexity mentioned above is the worst case total execution time of the entire experiment. This means the adversary complexity, defined as the worst case execution time of A plus the size of the code of the adversary A, in some fixed RAM model of computation (worst case means the maximum over A's coins or the answers returned in response to A's oracle queries), plus the time for other operations in the experiment, including the time for key generation and the computation of answers to oracle queries via execution of the encryption algorithm.

Another convention we make is that the length of a query M_0, M_1 to a left-or-right encryption oracle is defined as $|M_0|$. (We can assume without loss of generality that this equals $|M_1|$ since otherwise the oracle returns $\perp$ and so the query would be useless.) The total message length, which is the sum of the lengths of all oracle queries, is another parameter of interest. We say that the total message length is at most μ if it is so in the worst case, meaning across all coin tosses and answers to oracle queries in the experiment.

We consider an encryption scheme to be "secure against chosen-plaintext attack" if a "reasonable" adversary cannot obtain "significant" advantage, where reasonable reflects its resource

usage. The technical notion is called indistinguishability under chosen-ciphertext attack, denoted IND-CPA.

11.2.2 Security against chosen-ciphertext attack

Stories introducing chosen-ciphertext attack can be somewhat whimsical. One is about the so-called "lunchtime attack." Entity R goes to lunch while leaving his console accessible. For the short period of the lunch break, an adversary gets access to this console; when the lunch break is over, the adversary has to leave before it is discovered at the console by the legitimate user, returning from lunch. The access is such that the adversary cannot actually read the secret decryption key sk (imagine that sk is in protected hardware) but does have the capability of executing the algorithm $\mathcal{D}_{sk}(\cdot)$ on input any ciphertext of its choice. At that time if the adversary has in hand some ciphertext it wants to decrypt, it can certainly do so; there is nothing one can do to prevent that. However, it may be able to do even more. For example, perhaps there is some clever sequence of calls to $\mathcal{D}_{sk}(\cdot)$ via which the latter can be made to output sk itself. (These calls would not be made under normal execution of the algorithm on normal ciphertexts, but the adversary concocts weird ciphertexts that make the decryption routine do strange things.) Having sk means the adversary could decrypt traffic at any time in the future, even after the lunch break. Alternatively, the adversary is able to call $\mathcal{D}_{sk}(\cdot)$ on some inputs that result in the adversary's gaining some information that would enable it to decrypt some fraction of ciphertexts it might see later, after the lunch break, when it no longer has access to $\mathcal{D}_{sk}(\cdot)$. These are the eventualities we want to prevent.

This scenario is artificial enough that were it the only motivation, it would be natural to wonder whether it is really worth the trouble to design schemes to withstand chosen-ciphertext attack. But this is not the main motivation. The real motivation arises from gathering evidence that asymmetric encryption schemes secure against chosen-ciphertext attack are the desired and appropriate tool for use in many higher level protocols, for example protocols for authenticated session key exchange. There a party decrypts a random challenge message to prove its identity. This leaves it open to a chosen-ciphertext attack on the part of an adversary who sends ciphertexts in the guise of challenges and obtains their decryption. Were this attack to reveal the secret key, the adversary could impersonate the legitimate entity at a later date, since it would now itself possess the ability to decrypt the challenges sent by others.

Based on this and other such applications, we would like to design asymmetric encryption schemes that are secure against very strong kinds of chosen-ciphertext attack. To illustrate let's consider the following game. An adversary A is given a challenge ciphertext C and must output the corresponding plaintext to win the game. The adversary is given the public key pk under which C was created, and is also given access to the oracle $\mathcal{D}_{sk}(\cdot)$ allowing decryption under the secret key sk corresponding to pk. A trivial way for the adversary to win the game is to invoke its oracle on C. This triviality is the one thing disallowed. We allow the adversary to invoke $\mathcal{D}_{sk}(\cdot)$ on any input $C' \neq C$. Of course it may invoke the oracle multiple times; all the inputs provided to the oracle must however be different from C. If from the information so gathered the adversary can compute $\mathcal{D}_{sk}(C)$ then it wins the game.

This is a very strong form of chosen-ciphertext attack: the adversary can invoke the decryption oracle on any point other than the challenge. Again, one's first reaction might be that it is in fact ridiculously strong. How in any setting where I have some sort of decryption oracle access is it possible that I could not ask the query of my choice, yet be able to ask absolutely any other query? Indeed it is hard to imagine such a setting. Yet, this is the "right" attack model to consider for several reasons. One is that in proving the security of authenticated key exchange protocols that

use asymmetric encryption as discussed above, it is exactly security under such an attack that is required of the asymmetric encryption scheme. The other reasons is perhaps more fundamental. We have seen many times that it is difficult to anticipate the kinds of attacks that can arise. It is better to have an attack model that is clear and well defined even if perhaps stronger than needed, than to not have a clear model or have one that may later be found to be too weak.

We have already seen that inability to decrypt a challenge ciphertext is not evidence of security of a scheme, since one must also consider loss of partial information. In finalizing a notion of security against chosen-ciphertext attack one must take this into account too. This, however, we already know how to do, via left-or-right encryption oracles.

Definition 11.2.2 Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an asymmetric encryption scheme, let $b \in \{0,1\}$, and let A be an algorithm that has access to two oracles and returns a bit. We consider the following experiment:

Experiment $\mathbf{Exp}^{\text{ind-cca-}b}_{\mathcal{AE}}(A)$
  $(pk, sk) \xleftarrow{\$} \mathcal{K}$
  $b' \leftarrow A^{\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}_{sk}(\cdot)}(pk)$
  If A queried $\mathcal{D}_{sk}(\cdot)$ on a ciphertext previously returned by $\mathcal{E}_K(\mathbf{LR}(\cdot,\cdot,b))$
    then return 0
    else Return b'

The *ind-cca-advantage* of A is defined as

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(A) \;=\; \Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(A) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(A) = 1\right] \;. \quad \blacksquare$$

The conventions with regard to resource measures are the same as those used in the case of chosen-plaintext attacks.

We consider an encryption scheme to be "secure against chosen-ciphertext attack" if a "reasonable" adversary cannot obtain "significant" advantage, where reasonable reflects its resource usage. The technical notion is called indistinguishability under chosen-ciphertext attack, denoted IND-CCA.

11.3 One encryption query or many?

The adversary in our definitions is allowed to make many queries to its lr-encryption oracle. We gave it this power because it might be possible to expose weaknesses in the encryption scheme via an attack involving observing the encryptions of many related messages, chosen adaptively as a function of ciphertexts of previous messages. Indeed, it may be possible to achieve a higher advantage with more queries, but we show here that the gain is limited. Namely, an adversary making q_e lr-encryption oracle queries cannot achieve an advantage greater than q_e times that of an adversary making just one lr-encryption oracle query and having other resources comparable to that of the original adversary. This is true both under chosen-plaintext and chosen-ciphertext attack, as indicated in the following.

Theorem 11.3.1 Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an asymmetric encryption scheme. Let B be an ind-cpa adversary who makes at most q_e queries to its left-or-right encryption oracle. Then there exists an ind-cpa adversary A making at most one query to its left-or-right encryption oracle and such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(B) \;\leq\; q_e \cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A) \;. \tag{11.1}$$

Furthermore, the running time of A is that of B. Similarly, let B be an ind-cca adversary who makes at most q_e queries to its left-or-right encryption oracle. Then there exists an ind-cca adversary A making at most one query to its left-or-right encryption oracle and such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(B) \;\leq\; q_e \cdot \mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(A)\;. \tag{11.2}$$

Furthermore, the number of decryption oracle queries made by A is the same as made by B, and the running time of A is that of B. ▌

In a qualitative sense, this theorem can be interpreted as saying that an asymmetric encryption scheme secure against adversaries making just one lr-encryption query is also secure against adversaries making many lr-encryption queries. This will simplify later analyses by allowing us to focus on adversaries that make only one lr-encryption query.

An important element making this result possible is that in an asymmetric encryption scheme, an adversary can itself encrypt any message it wants, because it has the public (encryption) key. In the symmetric setting, the adversary cannot directly encrypt a message, but may only do so via an oracle that holds the key. An analogue of the above is true in the symmetric setting, but requires that the adversary be provided not only with an lr-encryption oracle but also with an encryption oracle.

Proof of Theorem 11.3.1: The statement corresponding to Equation (11.1) follows from the statement corresponding to Equation (11.2) by considering an ind-cca adversary who makes no queries to its decryption oracle, so we need only prove the statement corresponding to Equation (11.2).

We will use what's called a "hybrid argument". We will associate to B a sequence of experiments

$$\mathbf{Exp}^0_{\mathcal{AE}}(B)\;,\;\; \mathbf{Exp}^1_{\mathcal{AE}}(B)\;,\;\; \ldots\;,\;\; \mathbf{Exp}^q_{\mathcal{AE}}(B) \tag{11.3}$$

such that, if we let

$$P(i) \;=\; \Pr\left[\mathbf{Exp}^i_{\mathcal{AE}}(B) = 1\right]$$

for $i \in \{0, 1, \ldots, q\}$, then it will be the case that

$$P(0) \;=\; \Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(B) = 1\right] \tag{11.4}$$

$$P(q) \;=\; \Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(B) = 1\right]\;. \tag{11.5}$$

In other words, the first and last experiments in our sequence will correspond to the world 0 and world 1 experiments, respectively, in Definition 11.2.1. If so, Definition 11.2.1 tells us that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(B) \;=\; P(q) - P(0)\;.$$

Now comes a trick. We consider the sum

$$\sum_{i=1}^{q-1} [P(i) - P(i)]\;.$$

Its value, of course, is 0. Hence, from the above,

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(B) &= P(q) - P(0) \\
&= P(q) + \sum_{i=1}^{q-1} [P(i) - P(i)] - P(0) \\
&= \sum_{i=1}^{q} P(i) - \sum_{i=0}^{q-1} P(i)\;.
\end{aligned}$$

Oracle $\mathcal{HE}^i_{pk}(M_0, M_1)$	Experiment $\mathbf{Exp}^i_{\mathcal{AE}}(B)$
$j \leftarrow j + 1$	$(pk, sk) \xleftarrow{\$} \mathcal{K}$
If $j \leq i$	$d \leftarrow B^{\mathcal{HE}^i_{pk}(\cdot,\cdot),\mathcal{D}_{sk}(\cdot)}(pk)$
then $C \xleftarrow{\$} \mathcal{E}_{pk}(M_1)$	Return d
else $C \xleftarrow{\$} \mathcal{E}_{pk}(M_0)$	
EndIf	
Return C	

Adversary $A^{\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}_{sk}(\cdot)}(pk)$
$j \leftarrow 0$; $I \xleftarrow{\$} \{1, \ldots, q\}$
Subroutine $\mathcal{OE}(M_0, M_1)$
$j \leftarrow j + 1$
If $j < I$ then $C \xleftarrow{\$} \mathcal{E}_{pk}(M_1)$ EndIf
If $j = I$ then $C \xleftarrow{\$} \mathcal{E}_{pk}(\mathbf{LR}(M_0, M_1, b))$ EndIf
If $j > I$ then $C \xleftarrow{\$} \mathcal{E}_{pk}(M_0)$ EndIf
Return C
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot),\mathcal{D}_{sk}(\cdot)}(pk)$
Return d

Figure 11.1: Hybrid oracles and experiments related to the construction of ind-cca adversary A in the proof of Theorem 11.3.1.

We will now construct ind-cca-adversary A so that

$$\Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(A) = 1\right] = \frac{1}{q} \cdot \sum_{i=1}^{q} P(i) \tag{11.6}$$

$$\Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(A) = 1\right] = \frac{1}{q} \cdot \sum_{i=0}^{q-1} P(i)\ . \tag{11.7}$$

Then, from the above we would have

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(A) = \frac{1}{q} \cdot \mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(B)\ .$$

Re-arranging terms, we get Equation (11.2).

We now specify the "hybrid" experiments of Equation (11.3) in such a way that Equations (11.4) and (11.5) are true and we are able to construct adversary A such that Equations (11.6) and (11.7) are true.

We associate to any $i \in \{0, \ldots, q\}$ an oracle and an experiment, as indicated in Fig. 11.1. The oracle associated to i is stateful, maintaining a counter j that is initialized to 0 by the overlying experiment and is incremented by the oracle each time the latter is invoked.

Now, observe that oracles $\mathcal{HE}^0_{pk}(\cdot,\cdot)$ and $\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,0))$ are equivalent, meaning that on any inputs, their responses are identically distributed. Similarly, oracles $\mathcal{HE}^q_{pk}(\cdot,\cdot)$ and $\mathcal{E}_{pk}(\mathbf{LR}(\cdot,\cdot,1))$ are equivalent. Hence, Equations (11.4) and (11.5) are true.

Adversary A is specified in Fig. 11.1. It begins by initializing a counter j to 0, and picking I at random from $\{1,\ldots,q\}$. It then defines a subroutine $\mathcal{OE}$. Finally A executes B, replacing the B's lr-encryption oracle with the subroutine $\mathcal{OE}$, and providing B a decryption oracle via A's own access to a decryption oracle.

We highlight that A's operation depends on the fact that it was provided the public encryption key as an input. This enables it to compute encryptions under this key directly, and it does so inside the subroutine. Had we not given A the public key, this construction would not be posible.

To complete the proof it suffices to justify Equations (11.6) and (11.7). Suppose A is in world 1, meaning the challenge bit b equals 1. Then subroutine $\mathcal{OE}$ encrypts the right message of its input pair the first I times it is called, and the left-message after that. One the other hand, if A is in world 0, meaning $b = 0$, subroutine $\mathcal{OE}$ encrypts the right message of its input pair the first $I-1$ times it is called, and the left message after that. Regarding I as a random variable taking values in $\{1,\ldots,q\}$, this means that for every $i \in \{1,\ldots,q\}$ we have

$$\begin{aligned}
\Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(A) = 1 \mid I = i\right] &= P(i) \\
\Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(A) = 1 \mid I = i\right] &= P(i-1)\ .
\end{aligned}$$

Since the random variable I is uniformly distributed in the range $\{1,\ldots,q\}$ we have

$$\begin{aligned}
\Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(A) = 1\right] &= \sum_{i=1}^{q} \Pr\left[\mathbf{Exp}^{\text{ind-cca-1}}_{\mathcal{AE}}(A) = 1 \mid I = i\right] \cdot \Pr\left[I = i\right] \\
&= \sum_{i=1}^{q} P(i) \cdot \frac{1}{q}\ .
\end{aligned}$$

This justifies Equation (11.6). Similarly,

$$\begin{aligned}
\Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(A) = 1\right] &= \sum_{i=1}^{q} \Pr\left[\mathbf{Exp}^{\text{ind-cca-0}}_{\mathcal{AE}}(A) = 1 \mid I = i\right] \cdot \Pr\left[I = i\right] \\
&= \sum_{i=1}^{q} P(i-1) \cdot \frac{1}{q} \\
&= \sum_{i=0}^{q-1} P(i) \cdot \frac{1}{q}\ ,
\end{aligned}$$

which justifies Equation (11.7). This concludes the proof. ▌

11.4 Hybrid encryption

Before we present constructions of asymmetric encryption schemes, it is useful to get some idea of the context in which they are used.

Given an asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$, one rarely encrypts data directly with it. Rather, to encrypt M under a public key pk of this scheme, we first pick a random key

K for a symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$, encrypt K under pk via the asymmetric scheme to get a ciphertext C^a, encrypt M under K via the symmetric scheme to get a ciphertext C^s, and transmit (C^a, C^s). This is called hybrid encryption.

More precisely, hybrid encryption is a transform that given any asymmetric encryption scheme and any symmetric encryption scheme associates to them a new asymmetric encryption scheme:

Scheme 11.4.1 Let $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$ be an asymmetric encryption scheme, and let $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a stateless symmetric encryption scheme such that $\mathsf{Keys}(\mathcal{SE}) \subseteq \mathsf{Plaintexts}(pk)$ for every pk that might be output by $\mathcal{K}^a$. The hybrid encryption scheme associated to $\mathcal{AE}, \mathcal{SE}$ is the asymmetric encryption scheme $\overline{\mathcal{AE}} = (\mathcal{K}^a, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ whose key-generation algorithm is the same as that of $\mathcal{AE}$ and whose encryption and decryption algorithms are defined as follows:

Algorithm $\overline{\mathcal{E}}_{pk}(M)$	Algorithm $\overline{\mathcal{D}}_{sk}(C)$
$K \xleftarrow{\$} \mathcal{K}^s$; $C^s \xleftarrow{\$} \mathcal{E}^s_K(M)$	Parse C as (C^a, C^s)
If $C^s = \perp$ then return $\perp$	$K \leftarrow \mathcal{D}^a_{sk}(C^a)$
$C^a \xleftarrow{\$} \mathcal{E}^a_{pk}(K)$; $C \leftarrow (C^a, C^s)$	If $K = \perp$ then return $\perp$
Return C	$M \leftarrow \mathcal{D}^s_K(C^s)$
	Return M

Under this hybrid encryption scheme, one can (asymmetrically) encrypt any message M that is in the plaintext-space of the underlying symmetric encryption scheme. ▮

Hybrid encryption is used for numerous reasons. The principal one is cost. The number-theoretic operations underlying common asymmetric encryption schemes are computationally costly relative to the operations on block ciphers that underly common symmetric encryption schemes. In practice one wants to minimize the amount of data to which these number-theoretic operations are applied. Accordingly, rather than encrypt the possibly long message M directly under pk via the given asymmetric scheme, one uses hybrid encryption. The costly number-theoretic operations are thus applied only to data whose length k is fixed and not dependent on the length of M.

This context tells us that when we design asymmetric encryption schemes, we can typically assume that the message space consists of short strings. This will facilitate our constructions.

However, before we adopt the hybrid encryption paradigm we need to know that it "works," meaning that it is secure. In assessing the strength of hybrid encryption, we use as usual the provable-security philosophy and approach. A hybrid encryption scheme is built from two components: a base asymmetric encryption scheme and a base symmetric encryption scheme. The appropriate question to ask is whether the assumed security of the components suffices to guarantee security of the hybrid scheme based on them. It turns out that it does, and moreover for security under both chosen-plaintext and chosen-ciphertext atttacks. Theorem 11.4.2 below addresses the first case, and Theorem 11.4.3 the second. (Although the latter implies the former, we state and prove them separately because the proof has some delicate issues and ideas and is best understood via an incremental approach.)

Theorem 11.4.2 Let $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$ be an asymmetric encryption scheme, let $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a stateless symmetric encryption scheme such that

$$\mathsf{Keys}(\mathcal{SE}) \subseteq \mathsf{Plaintexts}(pk)$$

for every pk that might be output by $\mathcal{K}^a$, and let $\overline{\mathcal{AE}} = (\mathcal{K}^a, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ be the hybrid encryption scheme associated to $\mathcal{AE}, \mathcal{SE}$ as per Scheme 11.4.1. Let k denote the length of keys output by $\mathcal{K}^s$. Let B be

an ind-cpa-adversary attacking $\overline{\mathcal{AE}}$. Then there exist ind-cpa adversaries $A_{00,01}, A_{11,10}$ attacking $\mathcal{AE}$, and an adversary A attacking $\mathcal{SE}$, such that

$$\begin{aligned}&\mathbf{Adv}^{\text{ind-cpa}}_{\overline{\mathcal{AE}}}(B)\\ &\quad\leq\quad \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A_{00,01}) + \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A_{11,10}) + \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A)\ .\end{aligned} \tag{11.8}$$

Furthermore, suppose B had time complexity at most t, made at most q queries to its left-or-right encryption oracle, these totalling at most μ bits in length. Then $A_{00,01}, A_{11,10}$ each have time-complexity at most t and make at most q left-or-right encryption oracle queries, each query being k bits long. Also A has time-complexity at most t, and makes only one query to its left-or-right encryption oracle. ▮

The qualitative interpretation of Theorem 11.4.2 is that if $\mathcal{AE}$ and $\mathcal{SE}$ are each assumed to be secure against chosen-plaintext attack, then $\overline{\mathcal{AE}}$ is also secure against chosen-plaintext attack. On the quantitative front, note that the advantage of $\overline{\mathcal{AE}}$ against an attack involving q lr-encryption queries is upper bounded as a function of the advantage of $\mathcal{SE}$ against an attack involving only a *single* lr-encryption query. This means that the symmetric encryption scheme used may be very weak and yet the hybrid asymmetric encryption scheme will be secure. For example, the encryption algorithm of the symmetric encryption scheme could apply a pseudorandom bit generator to the key to get an output of $|M|$ bits and XOR this with the message to get the ciphertext. In particular, the symmetric encryption scheme could be deterministic.

Proof of Theorem 11.4.2: These constructions are not as straightforward as some we have seen in the past. We will need to "isolate" the asymmetric and symmetric components of $\overline{\mathcal{AE}}$ in such a way that an attack on this scheme can be broken down into attacks on the component schemes. To do this we will use a hybrid argument. We will associate to B a sequence of experiments

$$\mathbf{Exp}^{00}_{\overline{\mathcal{AE}}}(B)\ ,\ \ \mathbf{Exp}^{01}_{\overline{\mathcal{AE}}}(B)\ ,\ \ \mathbf{Exp}^{11}_{\overline{\mathcal{AE}}}(B)\ ,\ \ \mathbf{Exp}^{10}_{\overline{\mathcal{AE}}}(B) \tag{11.9}$$

such that, if we let

$$P(\alpha,\beta)\ =\ \Pr\left[\mathbf{Exp}^{\alpha\beta}_{\overline{\mathcal{AE}}}(B)=1\right] \tag{11.10}$$

for bits $\alpha,\beta\in\{0,1\}$, then it will be the case that

$$P(1,0) \quad=\quad \Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\overline{\mathcal{AE}}}(B)=1\right] \tag{11.11}$$

$$P(0,0) \quad=\quad \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\overline{\mathcal{AE}}}(B)=1\right]\ . \tag{11.12}$$

In other words, the first and last experiments in our sequence will correspond to the world 0 and world 1 experiments, respectively, in Definition 11.2.1. If so, Definition 11.2.1 tells us that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\overline{\mathcal{AE}}}(B)\ =\ P(1,0)-P(0,0)\ .$$

Now comes a trick. We throw into the expression $P(1,0)-P(0,0)$ a bunch of extra terms that sum to zero and hence don't change the value of the expression, and then we regroup, like this:

$$\begin{aligned}&P(1,0)-P(0,0)\\ &\quad=\quad P(1,0)-P(1,1)+P(1,1)-P(0,1)+P(0,1)-P(0,0)\\ &\quad=\quad [P(1,0)-P(1,1)]+[P(1,1)-P(0,1)]+[P(0,1)-P(0,0)]\ .\end{aligned}$$

Oracle $\mathcal{HE}^{00}_{pk}(M_0, M_1)$ $K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$ $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_0})$ If $C^s = \perp$ then return $\perp$ $C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_0})$ $C \leftarrow (C^a, C^s)$ Return C	Oracle $\mathcal{HE}^{01}_{pk}(M_0, M_1)$ $K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$ $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_0})$ If $C^s = \perp$ then return $\perp$ $C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_1})$ $C \leftarrow (C^a, C^s)$ Return C
Oracle $\mathcal{HE}^{11}_{pk}(M_0, M_1)$ $K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$ $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_1})$ If $C^s = \perp$ then return $\perp$ $C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_1})$ $C \leftarrow (C^a, C^s)$ Return C	Oracle $\mathcal{HE}^{10}_{pk}(M_0, M_1)$ $K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$ $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_1})$ If $C^s = \perp$ then return $\perp$ $C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_0})$ $C \leftarrow (C^a, C^s)$ Return C

Figure 11.2: Hybrid lr-encryption oracles used in the proof of Theorem 11.4.2.

We have now written the ind-cpa-advantage of B as a sum of the differences that adjacent experiments in our experiment sequence return 1. We will then construct the adversaries $A_{01,00}, A, A_{10,11}$ such that

$$P(0,1) - P(0,0) \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A_{01,00}) \tag{11.13}$$

$$P(1,1) - P(0,1) \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A) \tag{11.14}$$

$$P(1,0) - P(1,1) \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A_{10,11})\ . \tag{11.15}$$

Equation (11.8) follows.

The template above is pretty generic. What we need to do now is to actually specify the "hybrid" experiments of Equation (11.9) in such a way that Equations (11.11)–(11.12) are true and we are able to construct adversaries $A_{01,00}, A, A_{10,11}$ such that Equations (11.13)–(11.15) are true.

Recall that B has access to an oracle that takes input a pair of messages and returns a ciphertext. In the experiments of Definition 11.2.1 that define the ind-cpa-advantage of B, this oracle is either $\overline{\mathcal{E}}_{pk}(\mathbf{LR}(\cdot,\cdot,1))$ or $\overline{\mathcal{E}}_{pk}(\mathbf{LR}(\cdot,\cdot,0))$, depending on the world in which B is placed. Our hybrid experiments will involve executing B not only with these oracles, but with others that we will define. Specifically, we will define a sequence of oracles

$$\mathcal{HE}^{00}_{pk}(\cdot,\cdot)\ ,\ \ \mathcal{HE}^{01}_{pk}(\cdot,\cdot)\ ,\ \ \mathcal{HE}^{11}_{pk}(\cdot,\cdot)\ ,\ \ \mathcal{HE}^{10}_{pk}(\cdot,\cdot)\ . \tag{11.16}$$

Each oracle will take input a pair M_0, M_1 of messages and return a ciphertext. Now, to each pair α, β of bits, we associate the (α, β) hybrid experiment defined as follows:

Adversary $A_{01,00}^{\mathcal{E}_{pk}^a(\mathbf{LR}(\cdot,\cdot,b))}(pk)$
Subroutine $\mathcal{OE}(M_0, M_1)$
$K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$
$C^s \xleftarrow{\$} \mathcal{E}^s(K_0, M_0)$
If $C^s = \perp$ then return $\perp$
$C^a \xleftarrow{\$} \mathcal{E}_{pk}^a(\mathbf{LR}(K_0, K_1, b))$
Return (C^a, C^s)
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot)}(pk)$
Return d

Adversary $A_{10,11}^{\mathcal{E}_{pk}^a(\mathbf{LR}(\cdot,\cdot,b))}(pk)$
Subroutine $\mathcal{OE}(M_0, M_1)$
$K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$
$C^s \xleftarrow{\$} \mathcal{E}^s(K_0, M_1)$
If $C^s = \perp$ then return $\perp$
$C^a \xleftarrow{\$} \mathcal{E}_{pk}^a(\mathbf{LR}(K_1, K_0, b))$
Return (C^a, C^s)
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot)}(pk)$
Return d

Figure 11.3: Adversaries attacking $\mathcal{AE}$ constructed for the proof of Theorem 11.4.2.

Experiment $\mathbf{Exp}_{\mathcal{AE}}^{\alpha\beta}(B)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}^a$
$d \leftarrow B^{\mathcal{HE}_{pk}^{\alpha\beta}(\cdot,\cdot)}(pk)$
Return d

This defines our experiments in terms of the oracles, and, finally, the oracles themselves are specified in Fig. 11.2. Each *hybrid lr-encryption oracle* is paramterized by a pair (α, β) of bits and takes input a pair M_0, M_1 of messages. Examining the oracles, you will see that they are mostly identical, different only in the quantities that have been boxed. Each oracle picks not one but two keys K_0, K_1, independently at random, for symmetric encryption. It then encrypts M_α under K_0 via the symmetric encryption scheme to get a ciphertext C^s, and it encrypts K_β under the public key via the asymmetric encryption scheme to get a ciphertext C^a. It returns the pair (C^a, C^s).

Note oracles $\mathcal{HE}_{pk}^{00}(\cdot,\cdot)$ and $\mathcal{HE}_{pk}^{10}(\cdot,\cdot)$ do not actually use K_1. We have asked these oracles to pick K_1 only to highlight the common template underlying all four oracles.

Observe that oracle $\mathcal{HE}_{pk}^{00}(\cdot,\cdot)$ and oracle $\overline{\mathcal{E}}_{pk}(\mathbf{LR}(\cdot,\cdot,0))$ are equivalent in the sense that their responses to any particular query are identically distributed. Similarly oracle $\mathcal{HE}_{pk}^{10}(\cdot,\cdot)$ and oracle $\overline{\mathcal{E}}_{pk}(\mathbf{LR}(\cdot,\cdot,1))$ are equivalent. This means that Equations (11.11) and (11.12) are true, which is the first requirement of a successful hybrid argument.

The new hybrid lr-encryption oracles we introduce may seem rather bizarre at first since they do not necessarily return valid ciphertexts. For example, oracle $(0,1)$ will return a ciphertext (C^a, C^s) in which C^s is the encryption of M_0 under a key K_0, but C^a is not an encryption of K_0 as it ought to be under the definition of $\overline{\mathcal{AE}}$, but rather is the encryption of a random, unrelated key K_1. Thus, in the corresponding hybrid experiment, B is not getting the types of responses it "expects." But nonetheless, being an algorithm with access to an oracle, B will execute and eventually return a bit. The meaning of this bit may be unclear, but we will see that this does not matter.

Before constructing $A_{01,00}$ so that Equation (11.13) is true, let us try to explain the intuition. Consider hybrid lr-encryption oracles $(0,0)$ and $(0,1)$. Note that in both experiments, C^s is computed in exactly the same way. This means that the difference between $P(0,0)$ and $P(0,1)$ measures the

ability of the adversary to tell whether C^a encrypts the key underling C^s or not. This is something we can relate solely to the security of the base asymmetric encryption scheme.

Adversary $A_{01,00}$ attacking the base asymmetric encryption scheme $\mathcal{AE}$ is specified in Fig. 11.3. As per Definition 11.2.1, it has access to a lr-encryption oracle $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,b))$. Its strategy is to define a subroutine $\mathcal{OE}$ and then run B, using $\mathcal{OE}$ to reply to B's oracle queries. Subroutine $\mathcal{OE}$ takes input a pair M_0, M_1 of messages, picks a pair of keys for symmetric encryption, and finally returns a ciphertext which is computed using a call to the given oracle $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,b))$.

Consider $A_{01,00}$ in world 1, meaning its oracle is $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,1))$. In that case, the ciphertext C^a computed by subroutine $\mathcal{OE}(\cdot,\cdot)$ is an encryption of K_1, and thus subroutine $\mathcal{OE}(\cdot,\cdot)$ is equivalent to oracle $\mathcal{HE}^{01}_{pk}(\cdot,\cdot)$. On the other hand, when $A_{01,00}$ is in world 0, meaning its oracle is $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,0))$, the ciphertext C^a computed by subroutine $\mathcal{OE}(\cdot,\cdot)$ is an encryption of K_0, and thus subroutine $\mathcal{OE}(\cdot,\cdot)$ is equivalent to oracle $\mathcal{HE}^{00}_{pk}(\cdot,\cdot)$. Hence

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE}}(A_{01,00})=1\right] &= \Pr\left[\mathbf{Exp}^{01}_{\mathcal{AE}}(B)=1\right]\\ \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE}}(A_{01,00})=1\right] &= \Pr\left[\mathbf{Exp}^{00}_{\mathcal{AE}}(B)=1\right]\ .\end{aligned}$$

Subtracting, and remembering the notation of Equation (11.10), we get

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A_{01,00}) \;=\; P(0,1)-P(0,0)\ ,$$

which justifies Equation (11.13).

We leave to the reader the task of verifying Equation (11.15) based on the construction of $A_{11,10}$ given in Fig. 11.3, and now proceed to the construction of the ind-cpa adversary A attacking the base symmetric encryption scheme.

The intuition here is that the $(0,1)$ and $(1,1)$ hybrid experiments both compute C^a as an encryption of key K_1, but differ in which message they symmetrically encrypt under K_0, and thus the difference between $P(0,1)$ and $P(1,1)$ measures the ability of the adversary to tell which message C^s encrypts under K_0. This is something we can relate solely to the security of the base symmetric encryption scheme. The construction however will require a little more work, introducing another sequence of hybrid experiments. This time there will be $q+1$ of them,

$$\mathbf{Exp}^0_{\mathcal{AE}}(B)\ ,\ \ \mathbf{Exp}^1_{\mathcal{AE}}(B)\ ,\ \ \ldots\ ,\ \ \mathbf{Exp}^q_{\mathcal{AE}}(B)\ .$$

Again we associate to any $i \in \{0,\ldots,q\}$ an oracle and an experiment, as indicated in Fig. 11.4. The oracle associated to i is stateful, maintaining a counter j that is initially 0 and is incremented each time the oracle is invoked. The oracle behaves differently depending on how its counter j compares to its defining parameter i. If $j \le i$ it symmetrically encrypts, under K_0, the right message M_1, and otherwise it symmetrically encrypts, under K_0, the left message M_0. The asymmetric component C^a is always an encryption of K_1.

For $i = 0,\ldots,q$ we let

$$P(i) \;=\; \Pr\left[\mathbf{Exp}^i_{\mathcal{AE}}(B)=1\right]\ .$$

Now, suppose $i=0$. In that case, the value C^s computed by oracle $\mathcal{HE}^i_{pk}(\cdot,\cdot)$ on input M_0, M_1 is a symmetric encryption of M_0 regardless of the value of the counter j. This means that oracles $\mathcal{HE}^0_{pk}(\cdot,\cdot)$ and $\mathcal{HE}^{01}_{pk}(\cdot,\cdot)$ are equivalent. Similarly, oracles $\mathcal{HE}^q_{pk}(\cdot,\cdot)$ and $\mathcal{HE}^{11}_{pk}(\cdot,\cdot)$ are equivalent. Hence

$$P(0,1)=P(0) \qquad \text{and} \qquad P(1,1)=P(q)\ .$$

Oracle $\mathcal{HE}^i_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$
If $j \le i$
then $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_1})$
else $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_0})$
EndIf
If $C^s = \perp$ then return $\perp$
$C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_1})$
$C \leftarrow (C^a, C^s)$
Return C

Experiment $\mathbf{Exp}^i_{\overline{\mathcal{AE}}}(B)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}^a$
$d \leftarrow B^{\mathcal{HE}^i_{pk}(\cdot,\cdot)}(pk)$
Return d

Adversary $A^{\mathcal{E}^s_K(\mathbf{LR}(\cdot,\cdot,b))}$
$(pk, sk) \xleftarrow{\$} \mathcal{K}^a$; $j \leftarrow 0$; $I \xleftarrow{\$} \{1, \ldots, q\}$
Subroutine $\mathcal{OE}(M_0, M_1)$
$j \leftarrow j + 1$
$K_0 \xleftarrow{\$} \mathcal{K}^s$; $K_1 \xleftarrow{\$} \mathcal{K}^s$
If $j < I$ then $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_1})$ EndIf
If $j = I$ then $C^s \xleftarrow{\$} \mathcal{E}^s_K(\mathbf{LR}(M_0, M_1, b))$ EndIf
If $j > I$ then $C^s \xleftarrow{\$} \mathcal{E}^s(K_0, \boxed{M_0})$ EndIf
If $C^s = \perp$ then return $\perp$
$C^a \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_1})$
Return (C^a, C^s)
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot)}(pk)$
Return d

Figure 11.4: Hybrid oracles and experiments related to the construction of ind-cpa adversary A in the proof of Theorem 11.4.2.

So

$$\begin{aligned} & P(1,1) - P(0,1) \\ &= P(q) - P(0) \\ &= P(q) - P(q-1) + P(q-1) - \cdots - P(1) + P(1) - P(0) \\ &= \sum_{i=1}^{q} [P(i) - P(i-1)] \,. \end{aligned} \tag{11.17}$$

Our ind-cpa adversary A attacking the base symmetric encryption scheme $\mathcal{SE}$ is depicted in Fig. 11.4. It gets a lr-encryption oracle $\mathcal{E}^s_K(\mathbf{LR}(\cdot,\cdot,b))$ based on a hidden key K and challenge

bit b. It picks a pair of public and secret keys by running the key-generation algorithm of the asymmetric encryption scheme. It then picks an index i at random, and initializes a counter j to 0. Next it defines a subroutine $\mathcal{OE}(\cdot,\cdot)$ that takes input a pair of messages and returns a ciphertext, and runs B, replying to the latter's oracle queries via the subroutine. The subroutine increments the counter j at each call, and computes the symmetric component C^s of the ciphertext differently depending on how the counter j compares to the parameter i. In one case, namely when $j = i$, it computes C^s by calling the given $\mathcal{E}^s_K(\mathbf{LR}(\cdot,\cdot,b))$ oracle on inputs M_0, M_1. Notice that A makes only one call to its oracle, as required.

For the analysis, regard I as a random variable whose value is uniformly distributed in $\{1,\ldots,q\}$. Then notice that for any $i \in \{1,\ldots,q\}$

$$\begin{aligned}
\Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}}(A) = 1 \mid I = i\right] &= P(i) \\
\Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}}(A) = 1 \mid I = i\right] &= P(i-1)\ .
\end{aligned}$$

Thus

$$\begin{aligned}
&\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A) \\
&= \Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}}(A) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}}(A) = 1\right] \\
&= \sum_{i=1}^{q} \Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}}(A) = 1 \mid I = i\right] \cdot \Pr\left[I = i\right] \\
&\qquad - \sum_{i=1}^{q} \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}}(A) = 1 \mid I = i\right] \cdot \Pr\left[I = i\right] \\
&= \sum_{i=1}^{q} P(i) \cdot \Pr\left[I = i\right] \ - \ \sum_{i=1}^{q} P(i-1) \cdot \Pr\left[I = i\right] \\
&= \frac{1}{q} \cdot \sum_{i=1}^{q} P(i) - P(i-1) \\
&= \frac{1}{q} \cdot \left[P(1,1) - P(0,1)\right] .
\end{aligned}$$

In the last step we used Equation (11.17). Re-arranging terms, we get Equation (11.14). This completes the proof. ∎

We now proceed to the chosen-ciphertext attack case. The scheme itself is unchanged, but we now claim that if the base components are secure against chosen-ciphertext attack, then so is the hybrid encryption scheme.

Theorem 11.4.3 Let $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$ be an asymmetric encryption scheme, let $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a stateless symmetric encryption scheme such that

$$\mathsf{Keys}(\mathcal{SE}) \subseteq \mathsf{Plaintexts}(pk)$$

for every pk that might be output by $\mathcal{K}^a$, and let $\overline{\mathcal{AE}} = (\mathcal{K}^a, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ be the hybrid encryption scheme associated to $\mathcal{AE}, \mathcal{SE}$ as per Scheme 11.4.1. Let k denote the length of keys output by $\mathcal{K}^s$, and let

c denote the length of a ciphertext created by $\mathcal{E}^a$ on input a k-bit message. Let B be an ind-cpa-adversary attacking $\overline{\mathcal{AE}}$. Then there exist ind-cpa adversaries $A_{01,00}, A_{10,11}$ attacking $\mathcal{AE}$, and an adversary A attacking $\mathcal{SE}$, such that

$$\begin{aligned}&\mathbf{Adv}^{\text{ind-cca}}_{\overline{\mathcal{AE}}}(B)\\ &\quad\leq\quad \mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(A_{01,00}) + \mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}}(A_{10,11}) + \mathbf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(A)\ .\end{aligned}\tag{11.18}$$

Furthermore, suppose B had time complexity at most t, made at most q_e queries to its left-or-right encryption oracle, these totalling at most μ_e bits in length, and at most q_d queries to its decryption oracle, these totalling at most μ_d bits in length. Then $A_{00,01}, A_{11,10}$ each have time-complexity at most t and make at most q_e left-or-right encryption oracle queries, each query being k bits long, and at most q_d queries decryption oracle queries, each at most c bits long. Also A has time-complexity at most t, makes only one query to its left-or-right encryption oracle, and at most q_d queries to its decryption oracle. ▮

Proof of Theorem 11.4.3: We use a hybrid experiment template similar to the one in the proof of Theorem 11.4.2, but there are some tricky issues regarding decryption oracles. Let us try to highlight these before proceeding to the constructions.

Since B now has access to a decryption oracle, the (α, β) hybrid experiment of the proof of Theorem 11.4.2 will have to be enhanced to provide B with an oracle that plays the role of the decryption oracle that B expects. A natural first thought is to set this to the actual decryption oracle $\overline{\mathcal{D}}_{sk}(\cdot)$. Now, let us look ahead to the construction of $A_{01,00}$. Besides subroutine $\mathcal{OE}$ to replace B's lr-encryption oracle, $A_{01,00}$ will have to provide a subroutine $\mathcal{OD}$ to replace B's decryption oracle. This seems easy at first glance, because $A_{01,00}$, itself being a ind-cca-adversary, has access to a decryption oracle $\mathcal{D}_{sk}(\cdot)$ for the base asymmetric encryption scheme. Thus, it can simulate $\overline{\mathcal{D}}_{sk}(\cdot)$. The catch is in the rules of the game. Recall that $A_{01,00}$ is not allowed to call its decryption oracle on a ciphertext C^a that was previously returned by its own lr-encryption oracle. However, in attempting to simulate $\overline{\mathcal{D}}_{sk}(\cdot)$ using $\mathcal{D}_{sk}(\cdot)$, it might be forced to do so, because B might call $\overline{\mathcal{D}}_{sk}(\cdot)$ on a ciphertext (C^a, C^s) where a ciphertext of the form (C^a, X) *was* previously returned by B's lr-encryption oracle, but $X \neq C^s$. In that case, $A_{00,01}$ is stuck: how can it decrypt (C^a, C^s) without breaking the rules of its game?

To get around this problem we will enhance the hybrid experiments to provide B not with an actual decryption oracle, but with a fake one that we will define. Let us now proceed to the actual proof. To each pair α, β of bits, we associate the (α, β) hybrid experiment defined as follows:

$$\begin{array}{l}\text{Experiment } \mathbf{Exp}^{\alpha\beta}_{\overline{\mathcal{AE}}}(B)\\ \quad (pk, sk) \xleftarrow{\$} \mathcal{K}^a\ ;\ j \leftarrow 0\\ \quad d \leftarrow B^{\mathcal{HE}^{\alpha\beta}_{pk}(\cdot,\cdot),\mathcal{HD}_{sk}(\cdot,\cdot)}(pk)\\ \quad \text{Return } d\end{array}$$

The experiment initializes a counter j to 0, and then runs B, replacing B's lr-encryption oracle with a hybrid lr-encryption oracle, and B's decryption oracle with a hybrid decryption oracle. Note that the hybrid lr-encryption depends on (α, β) but the hybrid decryption oracle does not. However, the two oracles share state, in the form of counter j as well as quantities that are created and stored by the hybrid lr-encryption oracle and then accessed by the hybrid decryption oracle.

Oracle $\mathcal{HE}^{00}_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_0})$
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{0,j}})$
$C_j \leftarrow (C^a_j, C^s_j)$
Return C_j

Oracle $\mathcal{HE}^{01}_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_0})$
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{1,j}})$
$C_j \leftarrow (C^a_j, C^s_j)$
Return C_j

Oracle $\mathcal{HE}^{11}_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_1})$
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{1,j}})$
$C_j \leftarrow (C^a_j, C^s_j)$
Return C_j

Oracle $\mathcal{HE}^{10}_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_1})$
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{0,j}})$
$C_j \leftarrow (C^a_j, C^s_j)$
Return C_j

Oracle $\mathcal{HD}_{sk}(C)$
Parse C as (C^a, C^s)
$l \leftarrow \mathit{Find}(C^a; C^a_1, \ldots, C^a_j)$
If $l \neq 0$ then $M \leftarrow \mathcal{D}^s(K_{0,l}, C^s)$ EndIf
If $l = 0$ then $M \leftarrow \overline{\mathcal{D}}(sk, C)$ EndIf
Return M

Figure 11.5: Hybrid lr-encryption oracles, and hybrid decryption oracle, used in the proof of Theorem 11.4.3.

The hybrid lr-encryption oracles, shown in Fig. 11.5, are equivalent to the corresponding ones of Fig. 11.2 in the sense that on any inputs M_0, M_1, the output of the (α, β) hybrid lr-encryption oracle of Fig. 11.5 is distributed identically to the output of the (α, β) hybrid lr-encryption oracle of Fig. 11.2. However, the code of the hybrid lr-encryption oracles has been enhanced to do some extra internal book-keeping.

The hybrid decryption oracle invokes a subroutine *Find* that given a value T and a list $T_1, \ldots, T_j$ returns the smallest j such that $T = T_j$ if such a j exists, and 0 if $T \notin \{T_1, \ldots, T_j\}$. It uses this to see whether the asymmetric component of the ciphertext it is given to decrypt was previously returned by a partner (α, β) hybrid lr-encryption oracle. If not, it decrypts the given ciphertext via the decryption algorithm of scheme $\overline{\mathcal{AE}}$, using the secret key sk which it is given. Else, it decrypts the symmetric component of the ciphertext under the key $K_{0,j}$ chosen at the time the asymmetric component was first created.

Adversary $A_{01,00}^{\mathcal{E}_{pk}^a(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}_{sk}^a(\cdot)}(pk)$
$j \leftarrow 0$
Subroutine $\mathcal{OE}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C_j^s \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, M_0)$
If $C_j^s = \perp$ then return $\perp$
$C_j^a \xleftarrow{\$} \mathcal{E}_{pk}^a(\mathbf{LR}(K_{0,j}, K_{1,j}, b))$
Return (C_j^a, C_j^s)
End Subroutine
Subroutine $\mathcal{OD}(C)$
Parse C as (C^a, C^s)
$l \leftarrow \mathit{Find}(C^a; C_1^a, \ldots, C_j^a)$
If $l \neq 0$ then $M \leftarrow \mathcal{D}^s(K_{0,l}, C^s)$
If $l = 0$ then $M \leftarrow \overline{\mathcal{D}}(sk, C)$
Return M
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot),\mathcal{OD}(\cdot)}(pk)$
Return d

Adversary $A_{10,11}^{\mathcal{E}_{pk}^a(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}_{sk}^a(\cdot)}(pk)$
$j \leftarrow 0$
Subroutine $\mathcal{OE}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
$C_j^s \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, M_1)$
If $C_j^s = \perp$ then return $\perp$
$C_j^a \xleftarrow{\$} \mathcal{E}_{pk}^a(\mathbf{LR}(K_{1,j}, K_{0,j}, b))$
Return (C^a, C^s)
End Subroutine
Subroutine $\mathcal{OD}(C)$
Parse C as (C^a, C^s)
$l \leftarrow \mathit{Find}(C^a; C_1^a, \ldots, C_j^a)$
If $l \neq 0$ then $M \leftarrow \mathcal{D}^s(K_{0,l}, C^s)$
If $l = 0$ then $M \leftarrow \overline{\mathcal{D}}(sk, C)$
Return M
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot),\mathcal{OD}(\cdot)}(pk)$
Return d

Figure 11.6: Adversaries attacking $\mathcal{AE}$ constructed for the proof of Theorem 11.4.3.

As in the proof of Theorem 11.4.2, for any $\alpha, \beta \in \{0, 1\}$, we let

$$P(\alpha, \beta) \;=\; \Pr\left[\mathbf{Exp}_{\mathcal{AE}}^{\alpha\beta}(B) = 1\right] .$$

Observe that the hybrid decryption oracle is equivalent to $\overline{\mathcal{D}}_{sk}(\cdot)$ in the cases $(\alpha, \beta) \in \{(0, 0), (1, 0)\}$ because in these cases, the asymmetric component of the ciphertext produced by the hybrid lr-encryption oracle is an encryption of $K_{0,j}$. Thus we have

$$\begin{aligned} P(1,0) &= \Pr\left[\mathbf{Exp}_{\mathcal{AE}}^{\text{ind-cca-1}}(B) = 1\right] \\ P(0,0) &= \Pr\left[\mathbf{Exp}_{\mathcal{AE}}^{\text{ind-cca-0}}(B) = 1\right] . \end{aligned}$$

Following the proof of Theorem 11.4.2, our proof of Equation (11.18) is complete if we can construct adversaries $A_{01,00}, A, A_{10,11}$ such that

$$P(0,1) - P(0,0) \;\leq\; \mathbf{Adv}_{\mathcal{AE}}^{\text{ind-cca}}(A_{01,00}) \tag{11.19}$$

$$P(1,1) - P(0,1) \;\leq\; \mathbf{Adv}_{\mathcal{SE}}^{\text{ind-cca}}(A) \tag{11.20}$$

$$P(1,0) - P(1,1) \;\leq\; \mathbf{Adv}_{\mathcal{AE}}^{\text{ind-cca}}(A_{10,11}) . \tag{11.21}$$

The constructions of $A_{01,00}$ and $A_{10,11}$ are shown in Fig. 11.6. Each adversary runs B, replacing B's lr-encryption oracle with a subroutine $\mathcal{OE}$ and B's decryption oracle with a subroutine $\mathcal{OD}$.

Note the $\mathcal{OD}$ subroutine calls $\overline{\mathcal{D}}_{sk}(\cdot)$. It does not actually have sk but it can implement this by running the code of $\overline{\mathcal{D}}_{sk}(\cdot)$ shown in Scheme 11.4.1 and using its $\mathcal{D}^a_{sk}(\cdot)$ oracle.

We note that $A_{01,00}$ and $A_{10,11}$ are legal in the sense that they never query their decryption oracle $\mathcal{D}^a_{sk}(\cdot)$ on a ciphertext previously returned by their lr-encryption oracle $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,b)$. This is ensured by the definition of $\mathcal{OD}$, which, if given a ciphertext $C = (C^a, C^s)$ whose asymmetric component C^a was previously returned by $\mathcal{E}^a_{pk}(\mathbf{LR}(\cdot,\cdot,b)$, does not call $\mathcal{D}^a_{sk}(\cdot)$, but instead directly computes the symmetric decryption of C^s under a key $K_{0,j}$ satisfying $C^a_j = C^a$.

We leave to the reader to extend the arguments of the proof of Theorem 11.4.2 to verify that Equations (11.19) and (11.21) are true, and proceed to the construction of the ind-cca adversary A attacking the base symmetric encryption scheme. We associate to any $i \in \{0, \ldots, q\}$ the oracle and experiment defined in Fig. 11.7. The experiment shown in that figure initializes counter j and then runs B with the shown oracle and also with the hybrid decryption oracle $\mathcal{HD}_{sk}(\cdot)$ that we defined previously. The two oracles share state in the form of a counter j and as well as quantities that are created and stored by $\mathcal{HE}^i_{pk}(\cdot,\cdot)$ and then accessed by $\mathcal{HD}_{sk}(\cdot)$.

Our ind-cca adversary A attacking the base symmetric encryption scheme $\mathcal{SE}$ is depicted in Fig. 11.7. The novelty here, as compared to Fig. 11.4, is the subroutine $\mathcal{OD}$ defined by A. Note that it considers three cases for the value of l. In the second case, it computes a response by invoking A's given decryption oracle $\mathcal{D}^s_K(\cdot)$. In the third case it computes a response using the fact that it knows sk.

We must check that A is legal, meaning that it never queries its decryption oracle with a ciphertext C^s previously returned by its lr-encryption oracle. Suppose B makes decryption oracle query $C = (C^a, C^s)$. Our concern is that $C^s = C^s_I$. (The latter is the only ciphertext returned by A's lr-encryption oracle since A makes only one query to this oracle, and thus this is the only concern.) Let $l = \mathit{Find}(C^a; C^a_1, \ldots, C^a_j)$. If $l \neq I$ then A does not query its decryption oracle at all and thus certainly does not make an illegal query. So suppose $l = I$. This means $C^a = C^a_I$. However B is assumed to be legal, which implies $(C^a, C^s) \neq (C^a_I, C^s_I)$, so it must be that $C^s \neq C^s_I$ as desired.

The analysis of A is then analogous to the one in the proof of Theorem 11.4.2, and we omit the details. ▮

11.5 El Gamal scheme and its variants

Let G be a cyclic group with generator g, meaning $G = \{g^0, g^1, \ldots, g^{n-1}\}$, where $n = |G|$ is the order of G. Recall that the discrete exponentiation function is

$$\begin{array}{rccc} \mathrm{DExp}_{G,g}\colon & Z_n & \to & G \\ & x & \mapsto & g^x\ . \end{array}$$

The inverse of this function is the discrete logarithm function

$$\begin{array}{rccc} \mathrm{DLog}_{G,g}\colon & G & \to & Z_n \\ & X & \mapsto & x\ , \end{array}$$

where $x \in Z_n$ is the unique integer such that $g^x = X$ in G.

The discrete exponentiation function is conjectured to be one-way (meaning the discrete logarithm function is hard to compute) for some groups G. An example is the group $G = Z_p^*$ under

Oracle $\mathcal{HE}^i_{pk}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
If $j \leq i$
then $C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_1})$
else $C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_0})$
EndIf
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{1,j}})$
$C_j \leftarrow (C^a_j, C^s_j)$
Return C_j

Experiment $\mathbf{Exp}^i_{\overline{\mathcal{AE}}}(B)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}^a$
$d \leftarrow B^{\mathcal{HE}^i_{pk}(\cdot,\cdot),\mathcal{HD}_{sk}(\cdot)}(pk)$
Return d

Adversary $A^{\mathcal{E}^s_K(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}^s_K(\cdot)}$
$(pk, sk) \xleftarrow{\$} \mathcal{K}^a$; $j \leftarrow 0$; $I \xleftarrow{\$} \{1, \ldots, q\}$
Subroutine $\mathcal{OE}(M_0, M_1)$
$j \leftarrow j + 1$
$K_{0,j} \xleftarrow{\$} \mathcal{K}^s$; $K_{1,j} \xleftarrow{\$} \mathcal{K}^s$
If $j < I$ then $C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_1})$ EndIf
If $j = I$ then $C^s_j \xleftarrow{\$} \mathcal{E}^s_K(\mathbf{LR}(M_0, M_1, b))$ EndIf
If $j > I$ then $C^s_j \xleftarrow{\$} \mathcal{E}^s(K_{0,j}, \boxed{M_0})$ EndIf
If $C^s_j = \perp$ then return $\perp$
$C^a_j \xleftarrow{\$} \mathcal{E}^a(pk, \boxed{K_{1,j}})$
Return (C^a_j, C^s_j)
End Subroutine
Subroutine $\mathcal{OD}(C)$
Parse C as (C^a, C^s)
$l \leftarrow \mathit{Find}(C^a; C^a_1, \ldots, C^a_j)$
If $l \notin \{0, I\}$ then $M \leftarrow \mathcal{D}^s(K_{0,l}, C^s)$ EndIf
If $l = I$ then $M \leftarrow \mathcal{D}^s_K(C^s)$ EndIf
If $l = 0$ then $M \leftarrow \overline{\mathcal{D}}(sk, C)$ EndIf
Return M
End Subroutine
$d \xleftarrow{\$} B^{\mathcal{OE}(\cdot,\cdot),\mathcal{OD}(\cdot)}(pk)$
Return d

Figure 11.7: Hybrid oracles and experiments related to the construction of ind-cca adversary A in the proof of Theorem 11.4.3.

multiplication modulo p, where p is a large prime such that $p-1$ has a large prime factor. The size (order) of Z_p^* is $p-1$, so in this case $n = p-1$. In other words, exponents of g are in the range $0, 1, \ldots, p-2$.

Let us now assume G is some cyclic group in which the discrete logarithm problem is hard. We would like to use this assumption as the basis of an encryption scheme in the sense that, somehow, an adversary wanting to decrypt should be faced with solving a discrete logarithm problem. The basic idea is the following. Let the receiver's secret key by $x \in \mathbf{Z}_n$ and let its public key be $X = g^x \in G$. Note that computing the secret key given the public key involves computing a discrete logarithm and by assumption is hard. Now suppose a sender, in possession of X, picks $y \in \mathbf{Z}_n$ lets $Y = g^y \in G$, and sends Y to the receiver. At this point the receiver holds x, Y and the sender holds y, X. Consider the quantity $K = g^{xy} \in G$ and notice that

$$Y^x \;=\; (g^y)^x \;=\; \underbrace{g^{xy}}_{K} \;=\; (g^x)^y \;=\; X^y \;. \tag{11.22}$$

The sender can compute K as X^y since it knows y, X while the receiver can compute K via Y^x since it knows x, Y. The quantity K is thus a shared key. Having such a key, encryption of a message M is easy. Assuming that $M \in G$ is a group element, the sender computes $W = KM$ in G and transmits W to the receiver. The latter, having K, recovers M as WK^{-1}.

The above description might make it look like there are two steps, namely the sender first transmits Y and then W, but when we implement this as an encryption scheme, we simply merge these steps. The sender computes Y and W and the ciphertext it transmits is the pair (Y, W).

Now, what about security? The adversary is in possession of the public key X, and, via eavesdropping, will obtain the ciphertext (Y, W). The most direct attack is to attempt to compute $K = g^{xy}$. An adversary attempting to do this is faced with solving what we call the computationl Diffie-Hellman (CDH) problem: given X, Y compute g^{xy} where $X = g^x$ and $Y = g^y$. One approach to solving this is for the adversary to try either to find either x and then let $K = Y^x$, or to find y and let $K = X^y$. But finding x or y given X, Y involves computing discrete logarithms and is hard by assumption. However, even if the discrete logarithm problem is hard, we are not necessarily assured that computing K given X, Y is hard because there may be methods to do this that do not involve computing discrete logarithms. We do not know whether such methods exist or not, but we do know, empirically, that the CDH problem seems to be hard in numerous groups. Accordingly, the security of the scheme relies on this assumption rather than merely the assumption that the discrete logarithm problem is hard.

But this is a very rough view of the security considerations. When we examine security more closely we will see that for the schemes to achieve the kinds of strong, well-defined notions of security we have discussed above, the CDH assumption is not sufficient. But it is certainly the first cut at understanding the issues.

11.5.1 The El Gamal scheme

What we described informally above is the El Gamal encryption scheme. Let us now detail it and then look more closely at its security.

Scheme 11.5.1 Let G be a cyclic group of order n and let g be a generator of G. The *El Gamal encryption scheme* $\mathcal{AE}_{\rm EG} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to G, g is the asymmetric encryption scheme whose constituent algorithms are depicted below:

Algorithm $\mathcal{K}$	Algorithm $\mathcal{E}_X(M)$	Algorithm $\mathcal{D}_x((Y,W))$
$x \xleftarrow{\$} \mathbf{Z}_n$	If $M \notin G$ then return $\perp$	$K \leftarrow Y^x$
$X \leftarrow g^x$	$y \xleftarrow{\$} \mathbf{Z}_n$; $Y \leftarrow g^y$	$M \leftarrow WK^{-1}$
Return (X, x)	$K \leftarrow X^y$; $W \leftarrow KM$	Return M
	Return (Y, W)	

The plaintext-space associated to a public key $X \in G$ is G itself, and if M is not in this set then the encryption algorithm returns $\perp$. ▌

The quantities G, g are assumed to be chosen a priori and known to all parties. A typical example is $G = \mathbf{Z}_p^*$ where $p \geq 3$ is a prime. We have discussed in Section 9.3 how to find primes and generators and thus set up G, g in this case.

The first thing that should be verified about the El Gamal scheme is that decryption works correctly, meaning $\mathcal{D}_x(\mathcal{E}_X(M)) = M$ for all $M \in G$. This is true because of Equation (11.22), which says that the value K in both algorithms is indeed the same.

In common with several other algebraic schemes, in the natural formulation of the El Gamal scheme given above, the message is a group element. In practice we might prefer to think of our starting message as a string. In that case, we would encode the string as a group element before using the El Gamal scheme. For example if $G = \mathbf{Z}_p^*$ where p is a prime of length k (i.e. $2^{k-1} \leq p < 2^k$), the scheme could be viewed as enabling us to encrypt any binary string message m of length $k-1$. To do this, compute the integer whose binary representation is m and then adding one to it to get an integer M in the range $1, \ldots, 2^{k-1}$. This M beign in $\mathbf{Z}_p^*$ can be thought of as the message for the El Gamal scheme. From the group element returned by the decryption algorithm one can recover the corresponding string message in the obvious way. More generally, the message space can be viewed as any set of strings of size at most $|G|$, mapping these to group elements via some injective encoding function for the sake of encryption.

Now, we turn to security, concentrating first on security against chosen-plaintext attack. The first thing to consider is whether the adversary could recover the secret key x from the public key X. This however requires solving the discrete logarithm problem, which we are assuming is computationally intractable. Next we could consider the possibility of recovery of the plaintext M from a ciphertext (Y, W). The most obvious attack is for the adversary (given the public key X and a ciphertext (Y, W)) to try to compute the key K from X, Y, and then recover M via $M = [[WK^{-1} \bmod p]]^{-1}$. But trying to find K amounts to solving the CDH problem, which as we discussed is believed to be hard.

However, by now we know that it is naive to restrict security concerns to key recovery or even to recovery of plaintext from ciphertext. We must also address the possibility of loss of partial information about the plaintext. In other words, we should be asking whether the scheme meets the notion of IND-CPA we discussed above. Whether it does or not turns out to depend on the choice of group.

Before assessing IND-CPA, we need to clarify something. Recall that encryption is not, by our definition, required to hide the length of a message, captured by the fact that the left-or-right encryption oracle simply returns $\perp$ if fed a pair of messages of equal length. This leads us to ask what is the length of a message when the latter is a group element. As we said earlier, some encoding of group elements as strings is presumed. However, we insist that the strings corresponding to all elements of the group be of the same length, meaning encryption should not enable an adversary to distinguish the ciphertexts corresponding to any two group elements.

11.5.2 El Gamal in the group $\mathbf{Z}_p^*$

We first look at the case where $G = \mathbf{Z}_p^*$ for a prime $p \geq 3$ and show that in this case the scheme fails to be IND-CPA. The attacks rely on a little number theory from Chapter 9. Recall that the Legendre (also Jacobi) symbol $J_p(A)$ of $A \in \mathbf{Z}_p^*$ is 1 if A is a quadratic residue and -1 otherwise. We claim that given a ciphertext (Y, W) of the scheme above, we can compute $J_p(M)$. This is loss of information about M since a priori there is no reason that the Jacobi symbol of M should be known to an adversary.

We now explain how to compute $J_p(M)$ given an encryption (Y, W) of M under public key $X = g^x$. The scheme tells us that $W = KM$ where $K = g^{xy}$ and $g^y = Y$. We first note that by Proposition 9.20, $J_p(W) = J_p(KM) = J_p(K) \cdot J_p(M)$. This implies $J_p(M) = J_p(K) \cdot J_p(W)$. Now Proposition 9.21 tells us $J_p(K) = J_p(g^{xy})$ can be computed given $J_p(X)$ and $J_p(Y)$. Finally, Proposition 9.18 tells us that $J_p(X), J_p(Y), J_p(W)$ can all be computed in time cubic in the length of p. Putting it all together, $J_p(M)$ can be computed given X, Y, W in time cubic in the length of p. We now detail the attack.

Proposition 11.5.2 Let $p \geq 3$ be a prime and let $G = \mathbf{Z}_p^*$. Let g be a generator of G. Let $\mathcal{AE}_{\rm EG}$ be the El Gamal encryption scheme associated to G, g as per Scheme 11.5.1. Then there is an adversary A such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\rm EG}}(A) \;=\; 1\;.$$

Furthermore A makes only one query to its left-or-right encryption oracle and having running time $O(|p|^3)$ plus the time to perform some encoding related operations. ▮

Proof of Proposition 11.5.2: Adversary A has input a public key $X \in \mathbf{Z}_p^*$ and access to a left-or-right encryption oracle $\mathcal{E}_X(\mathbf{LR}(\cdot,\cdot,b))$, where $\mathcal{E}$ is the encryption algorithm of the scheme. (We regard p, g as fixed and known to all parties including the adversary.) Now here is the adversary:

Adversary $A^{\mathcal{E}_X(\mathbf{LR}(\cdot,\cdot,b))}(X)$
  $M_0 \leftarrow 1\,;\; M_1 \leftarrow g$
  $(Y, W) \xleftarrow{\$} \mathcal{E}_X(\mathbf{LR}(M_0, M_1, b))$
  If $X^{(p-1)/2} \equiv -1 \pmod p$ and $Y^{(p-1)/2} \equiv -1 \pmod p$)
    then $s \leftarrow -1$ else $s \leftarrow 1$
  EndIf
  If $W^{(p-1)/2} \equiv s \pmod p$ then return 0 else return 1 EndIf

Results in previous chapters Propositions 9.21 and 9.18 tell us that $s = J_p(K)$ where $K = g^{xy}$, $Y = g^y$ and $X = g^x$. By Proposition 9.20 and some basic algebra, we have

$$J_p(W) = J_p(KM_b^{-1}) = J_p(K) \cdot J_p(M_b^{-1}) = J_p(K) \cdot J_p(M_b) = s \cdot J_p(M_b)$$

where b is the challenge bit. Proposition 9.17 tells us that M_0 is a square (it equals g^0 and 0 is even) and M_1 is a non-square (it equals g^1 and 1 is odd). Now suppose we are in world 0, meaning $b = 0$. Then $J_p(M_b) = 1$ so $J_p(W) = s$ and thus A returns 0. On the other hand if we are in world 1, meaning $b = 1$, then $J_p(M_b) = -1$, so $J_p(W) \neq s$ and A returns 1. This means that

$$\begin{aligned}
\Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE}_{\rm EG}}(A) = 1\right] &= 1 \\
\Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE}_{\rm EG}}(A) = 1\right] &= 0\;.
\end{aligned}$$

Subtracting, we get

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\text{EG}}}(A) \;=\; 1$$

as desired. ▌

11.5.3 Chosen-ciphertext attacks on the El Gamal scheme

The El Gamal scheme is vulnerable to a chosen-ciphertext attack regardless of the choice of group G. An adversary can obtain the decryption of a given ciphertext by calling the decryption oracle on a different but related ciphertext. This leads to the following:

Proposition 11.5.3 Let G be a cyclic group and g a generator of G. Let $\mathcal{AE}_{\text{EG}}$ be the El Gamal encryption scheme associated to G, g as per Scheme 11.5.1. Then there is an adversary A such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AE}_{\text{EG}}}(A) \;=\; 1\ .$$

Furthermore A makes one query to its left-or-right encryption oracle, one query to its decryption oracle, and has running time the cost of a few exponentiations in G. ▌

Proof of Proposition 11.5.3: Adversary A that has input a public key $X \in G$ and access to two oracles: a left-or-right encryption oracle $\mathcal{E}_X(\mathbf{LR}(\cdot,\cdot,b))$ and a decryption oracle $\mathcal{D}_x(\cdot)$ where $g^x = X$. (Group G and generator g are fixed and known to all parties including the adversary, and $\mathcal{E}, \mathcal{D}$ are as in Scheme 11.5.1). It works as follows:

Adversary $A^{\mathcal{E}_X(\mathbf{LR}(\cdot,\cdot,b)),\mathcal{D}_x(\cdot)}(X)$
  Let M_0, M_1 be any two distinct elements of G
  $(Y, W) \xleftarrow{\$} \mathcal{E}_X(\mathbf{LR}(M_0, M_1, b))$
  $W' \leftarrow Wg$
  $M \leftarrow \mathcal{D}_x((Y, W'))$
  If $M = M_0 g$ then return 0 else return 1

The ciphertext (Y, W') is different from the ciphertext (Y, W) and thus the adversary is allowed to call its decryption oracle on (Y, W'). Let b denote the challenge bit and let $K = g^{xy}$ where $Y = g^y$. Then

$$M \;=\; \mathcal{D}_x((Y, W')) \;=\; K^{-1}W' \;=\; K^{-1}Wg \;=\; M_b g\ .$$

Thus the value returned by A is the bit b, meaning it has advantage 1. ▌

11.5.4 Security of El Gamal under the DDH assumption

In suitable groups, the El Gamal encryption scheme is secure against chosen-plaintext attack. The groups in question are those for which the DDH (Decision Diffie-Hellman) problem is hard. The problem was described in Section 10.1.4. Recall the problem is that the adversary is given g^x, g^y, g^z and must return a bit to indicate whether or not $g^z = g^{xy}$, where g is a generator of the underlying cyclic group G. If one can solve the CDH problem then one can solve the DDH problem by computing g^{xy} and testing whether it equals g^z, but it might be possible to solve the DDH problem without solving the CDH problem. Indeed, this is true in some groups such as

integers modulo a prime, as indicated by Proposition 10.5 meaning the DDH problem is easy in these groups. But there are choices of group for which the DDH problem is hard, in particular some groups of prime order such as the subgroup of quadratic residues of the group of integers modulo a prime (cf. Section 9.5).