第 11 章 公钥加密

11.1 公钥加密概述

公钥加密的引入标志着密码学的一场革命。在此之前，密码学家完全依赖**共享的秘密密钥**来实现私密通信。相比之下，公钥技术使得通信双方**无需事先约定任何秘密信息**即可进行私密通信。正如我们已经指出的，这是非常令人惊奇和反直觉的：这意味着房间两边只能通过大喊进行交流、没有任何初始秘密的两个人，可以以房间里任何其他人都无法知晓他们谈话内容的方式进行交流！

在私钥加密的环境中，通信双方约定一个秘密密钥，双方都可以使用该密钥进行加密和解密。公钥加密在这两个方面都是**非对称的**。其中一方（接收方）生成一对密钥 **(pk, sk)**，分别称为**公钥**（public key）和**私钥**（private key）。公钥用于发送方加密消息；接收方使用私钥解密得到的密文。

既然目标是避免通信双方需要事先会面以约定任何信息，那么发送方如何获取 **pk** 呢？从抽象层面来看，有两种方法可以实现。我们称接收方为**爱丽丝**（Alice），发送方为**鲍勃**（Bob）。

在第一种方法中，当爱丽丝得知鲍勃想与她通信时，她可以当场生成 **(pk, sk)**（假设她尚未生成），然后将 **pk 以明文形式**发送给鲍勃；鲍勃随后可以使用 **pk** 加密他的消息。我们强调，爱丽丝和鲍勃之间的信道可以是公开的，但假定是**经过身份验证的**（authenticated），这意味着对手不能修改爱丽丝发送给鲍勃的公钥（尤其不能将其替换为自己的密钥）。关于如何在**未经身份验证的信道**上分发公钥，请参阅第 12.7 节的讨论。

另一种方法是爱丽丝事先生成她的密钥 **(pk, sk)**，这与任何特定的发送方无关。（事实上，在密钥生成时，爱丽丝甚至不需要知道鲍勃想与她交谈，甚至不需要知道鲍勃的存在。）爱丽丝可以通过例如在她的网页上发布、放在她的名片上或放置在公共目录中来广泛传播她的公钥 **pk**。现在，任何希望与爱丽丝进行私密通信的人都可以查找她的公钥，并按上述方式进行。请注意，多个发送方可以使用相同的公钥 **pk** 多次与爱丽丝通信，用于加密他们的所有通信。

请注意，**pk** 本质上是**公开的**，因此在上述任何一种情况下，攻击者都很容易获取。在第一种情况下，窃听爱丽丝和鲍勃之间通信的对手会直接获得 **pk**；在第二种情况下，对手可以自己去查找爱丽丝的公钥。因此，公钥加密的安全性**不能依赖**于 **pk** 的保密性，而必须依赖于 **sk** 的保密性。因此，至关重要的是爱丽丝不能向任何人，包括发送方鲍勃，泄露她的私钥。

与私钥加密的比较

私钥加密与公钥加密之间最明显的区别可能是：前者假设**所有**密码密钥都必须完全保密，而后者只要求**私钥 sk** 保密。尽管这看起来只是一个小区别，但其后果是巨大的：在私钥环境中，通信双方必须以某种方式共享秘密密钥，而不允许任何第三方知晓，而在公钥环境中，公钥可以通过公共信道从一方发送给另一方，而不会损害安全性。对于在房间里大喊或更现实地通过公共网络（如电话线或互联网）进行通信的各方，公钥加密是唯一的选择。

另一个重要的区别是，私钥加密方案使用**相同的密钥**进行加密和解密，而公钥加密方案使用**不同的密钥**进行每项操作。也就是说，公钥加密本质上是**非对称的**。公钥设置中的这种非对称性意味着发送方和接收方的角色**不能互换**，而它们在私钥设置中是可互换的：单个密钥对只允许**单向**通信。（双向通信可以通过多种方式实现；关键是公钥加密方案的单次调用会强制区分充当**接收方**的一个用户和充当**发送方**的其他用户。）此外，公钥加密方案的单个实例能够允许多个发送方与单个接收方进行私密通信，这与私钥情况形成对比，私钥情况下，两个方之间共享的秘密密钥仅允许这两个方之间进行私密通信。

总结和阐述前面的讨论，我们看到公钥加密相对于私钥加密具有以下优势：

- 公钥加密**（在一定程度上）解决了密钥分发问题**，因为通信双方无需在通信前秘密共享密钥。即使它们之间的所有通信都被监控，双方仍可以秘密通信。
- 当单个接收方与 N 个发送方通信时（例如，在线商家处理来自多个购买者的信用卡订单），接收方存储**单个私钥 sk** 比共享、存储和管理 N 个不同的秘密密钥（即每个发送方一个）要方便得多。事实上，在使用公钥加密时，在密钥生成时**无需知道**潜在发送方的数量和身份。这为“开放系统”提供了巨大的灵活性。

公钥加密方案允许任何人充当发送方的事实，当接收方只想接收来自**特定个人**的消息时，可能会成为一个缺点。在这种情况下，经过身份验证的（私钥）加密方案将是比公钥加密更好的选择。

公钥加密的主要缺点是它比私钥加密**慢**大约 2 到 3 个数量级。在资源严重受限的设备（如智能卡或射频识别（RFID）标签）中实现公钥加密可能是一个挑战。即使是台式计算机执行密码操作，每秒执行数千次此类操作（如在线商家处理信用卡交易的情况）也可能令人望而却步。因此，当私钥加密是一个选项时（即如果双方可以事先安全地共享一个密钥），通常应该使用它。

事实上，正如我们将在 11.3 节中看到的那样，在公钥设置中会使用私钥加密来提高长消息（的公钥）加密的效率。因此，对私钥加密的透彻理解对于理解公钥加密在实践中是如何实现的至关重要。

公钥的安全分发

到目前为止，我们所有的讨论都隐含地假设对手是**被动的**；也就是说，对手只窃听发送方和接收方之间的通信，但不会积极干扰通信。如果对手有能力篡改诚实双方之间的所有通信，并且这些诚实双方事先没有共享任何密钥，那么私密性就根本无法实现。例如，如果接收方爱丽丝将她的公钥 **pk** 发送给鲍勃，但对手将其替换为它自己的密钥 pk'（对手知道匹配的私钥 sk'），那么即使鲍勃使用 pk' 加密了他的消息，对手也将能够轻松恢复该消息（使用 sk'）。如果对手能够更改存储在某个公共目录中的爱丽丝的公钥的值，或者如果对手能够在公钥从公共目录传输到鲍勃时对其进行篡改，也会发生类似的攻击。如果爱丽丝和鲍勃事先没有共享任何信息，并且不愿意依赖某个相互信任的第三方，那么爱丽丝或鲍勃无法采取任何措施来阻止此类主动攻击，甚至无法知道此类攻击正在发生。

重要的是，本章对公钥加密的处理假设发送方能够获得接收方公钥的**合法副本**。（这将在我们提供的安全定义中隐含。）也就是说，我们假设**安全密钥分发**。之所以做出这个假设，并不是因为上面讨论的主动攻击不值得关注——事实上，它们代表着任何使用公钥加密的实际系统都必须处理的严重威胁。相反，做出这个假设是因为存在其他机制来防止主动攻击（例如，参见第 12.7 节），因此将安全公钥加密的研究与安全公钥分发的研究解耦是方便（且有用）的。

11.2 定义

我们首先定义公钥加密的**语法**。该定义与定义 3.7 非常相似，不同之处在于我们现在使用的是**不同的加密和解密密钥**，而不是只使用一个密钥。

定义 11.1 公钥加密方案是一个由**概率多项式时间算法**组成的**三元组 (Gen, Enc, Dec)**，满足以下条件：

1. **密钥生成算法 Gen** 以**安全参数** 1^n 为输入，并输出一对密钥 **(pk, sk)**。我们将其中第一个称为**公钥**，第二个称为**私钥**。为方便起见，我们假设 **pk** 和 **sk** 的长度都至少为 n，并且可以从 **pk, sk** 确定 n。
2. **加密算法 Enc** 以公钥 **pk** 和来自某个消息空间（可能取决于 **pk**）的消息 m 为输入。它输出一个密文 c，我们记为 $c \leftarrow Enc_{pk}(m)$。（展望未来，Enc 需要是**概率性**的才能实现有意义的安全性。）
3. **确定性解密算法 Dec** 以私钥 **sk** 和密文 c 为输入，输出一个消息 m 或表示失败的特殊符号 $\perp$。我们记为 $m := Dec_{sk}(c)$。

要求除了由 $Gen(1^n)$ 输出的 **(pk, sk)** 的**可忽略概率**外，对于任何（合法）消息 m，我们都有 $Dec_{sk}(Enc_{pk}(m)) = m$。

与私钥设置的重要区别在于，密钥生成算法 **Gen** 现在输出**两个密钥**而不是一个。公钥 **pk** 用于加密，而私钥 **sk** 用于解密。重申我们之前的讨论，**pk** 被假定为广泛分发的，以便任何人都可以为生成此密钥的一方加密消息，但 **sk** 必须由接收方保密，安全性才可能成立。

我们允许存在**可忽略的**解密错误概率，事实上，我们提出的一些方案将具有可忽略的错误概率（例如，如果需要选择一个素数，但以可忽略的概率选择了合数）。尽管如此，我们通常会忽略这个问题。

对于公钥加密的实际使用，我们希望消息空间是 $\{0,1\}^*$ 或 $\{0,1\}^n$（并且特别是独立于公钥）。虽然我们有时会描述使用某个消息空间 $\mathcal{M}$ 的加密方案，该消息空间不包含某个固定长度的所有位串（并且可能也取决于公钥），但在这种情况下，我们也会指定如何将位串编码为 $\mathcal{M}$ 的元素。此编码必须既能高效计算，又能高效可逆，以便接收方可以恢复被加密的位串。

11.2.1 抵抗选择明文攻击的安全性

我们首先引入公钥设置中与定义 3.8 “自然”对应的安全性概念。由于该定义（以及我们将看到的其他定义）的广泛动机已在第 3 章中给出，因此此处的讨论将相对简短，并将主要关注私钥设置和公钥设置之间的**差异**。

给定一个公钥加密方案 $\Pi = (Gen, Enc, Dec)$ 和一个对手 $\mathcal{A}$，考虑以下实验：

窃听不可区分性实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 **(pk, sk)**。
2. 给予对手 $\mathcal{A}$ **pk**， $\mathcal{A}$ 输出一对**等长**的消息 m_0, m_1，它们在消息空间中。
3. 选择一个均匀的比特 $b \in \{0,1\}$，然后计算密文 $c \leftarrow Enc_{pk}(m_b)$ 并将其给予 $\mathcal{A}$。我们称 c 为**挑战密文**。
4. $\mathcal{A}$ 输出一个比特 b'。如果 $b' = b$，则实验的输出为 1，否则为 0。如果 $b' = b$，我们说 $\mathcal{A}$ **成功**。

定义 11.2 如果对于所有**概率多项式时间**对手 $\mathcal{A}$，存在一个**可忽略函数** negl，使得

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

则公钥加密方案 $\Pi = (Gen, Enc, Dec)$ 在**窃听者**存在的情况下具有**不可区分加密**（indistinguishable encryptions in the presence of an eavesdropper）。

上述定义与定义 3.8 之间的主要区别在于，这里**给予了 $\mathcal{A}$ 公钥 pk**。此外，我们允许 $\mathcal{A}$ 根据此公钥选择其消息 m_0 和 m_1。这对于定义公钥加密的安全性至关重要，因为如前所述，我们

假设对手知道接收方的公钥。

给予对手 $\mathcal{A}$ 用于加密消息的公钥 **pk** 这一看似“微小的”修改具有巨大的影响：它实际上**免费**给予 $\mathcal{A}$ 访问**加密预言机**的能力。（加密预言机的概念在 3.4.2 节中解释。）这是因为对手，在给定 **pk** 的情况下，可以通过简单地计算 $Enc_{pk}(m)$ 来自行加密任何消息 m。（与往常一样，假设 $\mathcal{A}$ 知道算法 Enc。）结果是，定义 11.2 **等同于 CPA-安全性**（即抵抗**选择明文攻击**的安全性），其定义方式类似于定义 3.22，唯一的区别是在相应的实验中攻击者被给予了公钥。

因此我们有：

命题 11.3 如果公钥加密方案在窃听者存在的情况下具有不可区分加密，则它是 **CPA-安全**的。

这与私钥设置形成了对比，在私钥设置中，存在在窃听者存在的情况下具有不可区分加密但**在选择明文攻击下不安全**的方案（参见命题 3.20）。

接下来讨论紧随上述结果而来的、与私钥设置进一步的差异。

完美秘密公钥加密的不可能性。 完美秘密公钥加密可以类似于定义 2.3 进行定义，条件是基于窃听者的完整视图（即包括公钥）。等效地，可以通过扩展定义 11.2 来定义它，要求对于所有对手 $\mathcal{A}$（不仅是高效的），我们都有：

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1] = \frac{1}{2}$$

然而，与私钥设置相反，**完美秘密公钥加密是不可能**的，无论密钥有多长或消息空间有多小。事实上，一个**无界**的对手，在给定 **pk** 和通过 $c \leftarrow Enc_{pk}(m)$ 计算出的密文 c 时，可以以概率 1 确定 m。对此的证明留作练习 11.1。

确定性公钥加密的不安全性。 正如在私钥加密的背景下所指出的，**任何确定性加密方案都不能是 CPA-安全**的。这里也是如此：

定理 11.4 任何确定性公钥加密方案都不是 CPA-安全的。

由于定理 11.4 非常重要，因此值得进行更多讨论。该定理不是我们安全定义的“产物”，也不是表明我们的定义过于强硬。**确定性公钥加密方案在现实场景中容易受到实际攻击，并且永远不应该使用。** 原因是确定性方案不仅允许对手确定何时两次发送相同的消息（如在私钥设置中），而且还允许对手，如果被加密的**可能消息集很小**，则以概率 1 恢复消息。例如，考虑一位教授加密学生的成绩。在这里，窃听者知道每个学生的成绩必须是 {A, B, C, D, F} 之一。如果教授使用确定性公钥加密方案，窃听者可以通过加密所有可能的成绩并将结果与给定的密文进行比

较，从而快速确定任何学生的实际成绩。

尽管上述定理看似非常简单，但在很长一段时间内，许多现实世界的系统都是使用确定性公钥加密设计的。公平地说，当公钥加密引入时，概率加密的重要性尚未完全认识到。Goldwasser 和 Micali 的开创性工作，其中提出了（与）定义 11.2（等效的概念）并阐述了定理 11.4，标志着密码学领域的一个转折点。首次以正式定义固定直觉并以正确的方式看待事物的重要性（即使事后看来很简单）不应被低估。

11.2.2 多次加密

与第 3 章一样，重要的是要了解使用相同的密钥（在这种情况下，是相同的公钥）加密多条消息的影响。我们可以像定义 3.19 中那样，通过让对手输出两个明文列表来制定这种设置中的安全性。然而，出于 3.4.2 节中讨论的原因，我们选择使用一个定义，其中攻击者可以访问一个**“左或右”预言机** $LR_{pk,b}$，该预言机在输入一对等长消息 m_0, m_1 时，计算密文 $c \leftarrow Enc_{pk}(m_b)$ 并返回 c。允许攻击者根据需要多次查询该预言机，因此该定义对使用相同公钥加密多条（未知）消息时的安全性进行建模。

形式上，考虑为公钥加密方案 $\Pi = (Gen, Enc, Dec)$ 和对手 $\mathcal{A}$ 定义的以下实验：

LR-预言机实验 $\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 **(pk, sk)**。
2. 选择一个均匀的比特 $b \in \{0,1\}$。
3. 给予对手 $\mathcal{A}$ 输入 **pk** 和对 $LR_{pk,b}(\cdot,\cdot)$ 的预言机访问权限。
4. 对手 $\mathcal{A}$ 输出一个比特 b'。
5. 如果 $b' = b$，则实验的输出定义为 1，否则为 0。如果 $\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n) = 1$，我们说 $\mathcal{A}$ **成功**。

定义 11.5 如果对于所有概率多项式时间对手 $\mathcal{A}$，存在一个可忽略函数 negl，使得：

$$Pr[\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

则公钥加密方案 $\Pi = (Gen, Enc, Dec)$ 具有**不可区分多次加密**（indistinguishable multiple encryptions）。

我们将证明任何 CPA-安全的方案都会自动具有不可区分多次加密；也就是说，在公钥设置中，**单个消息加密的安全性蕴含多条消息加密的安全性**。这意味着我们可以证明某个方案相对于定义 11.2 的安全性（该定义更简单、更容易使用），并得出该方案满足定义 11.5 的结论，后者是

一个看似更强的定义，可以更准确地模拟公钥加密的实际使用。以下定理的证明将在下面给出。

定理 11.6 如果公钥加密方案 Π 是 CPA-安全的，那么它也具有**不可区分多次加密**。

私钥设置中也有类似的结果，但未作为定理 3.24 得到证明。

加密任意长度的消息。 定理 11.6 的一个直接推论是，一个针对固定长度消息的 CPA-安全公钥加密方案，可以推导出一个满足相同安全性概念的**任意长度消息**的公钥加密方案。我们在原始方案仅加密 1 位消息的极端情况下说明这一点。假设 $\Pi = (Gen, Enc, Dec)$ 是一个用于单比特消息的加密方案。我们可以构造一个具有消息空间 $\{0,1\}^*$ 的新方案 $\Pi' = (Gen, Enc', Dec')$，方法是如下定义 Enc'：

$$Enc'_{pk}(m) = Enc_{pk}(m_1), \ldots, Enc_{pk}(m_l), \quad \text{其中 } m = m_1 \cdots m_l.$$

（解密算法 Dec' 以显而易见的方式构造。）我们有：

主张 11.7 令 Π 和 Π' 如上。如果 Π 是 CPA-安全的，那么 Π' 也是 CPA-安全的。

该主张之所以成立，是因为我们可以将使用 Π' 加密消息 m 视为使用方案 Π 加密 l 条消息 $(m_1, \ldots, m_l)$。

术语说明。 我们已经介绍了公钥加密方案的三个安全性定义——在窃听者存在下的不可区分加密、CPA-安全性以及不可区分多次加密——它们都是**等价的**。遵循密码学文献中的惯例，我们将简单地使用术语 **“CPA-安全性”** 来指代满足这些安全性概念的方案。

*定理 11.6 的证明

定理 11.6 的证明相当复杂。因此，在转向细节之前，我们提供一些直觉。对于这种直觉讨论，我们假设为了简化，$\mathcal{A}$ 在实验 $\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n)$ 中只进行了**两次**对 LR 预言机的调用。（在完整的证明中，调用次数是任意的。）

固定一个任意的 PPT 对手 $\mathcal{A}$ 和一个 CPA-安全的公钥加密方案 Π，并考虑一个实验 $\mathrm{PubK}^{\mathrm{LR-cpa2}}_{\mathcal{A},\Pi}(n)$，其中 $\mathcal{A}$ 只能对 LR 预言机进行两次查询。将 $\mathcal{A}$ 对预言机进行的查询表示为 $(m_{1,0}, m_{1,1})$ 和 $(m_{2,0}, m_{2,1})$；请注意，第二对消息可能取决于 $\mathcal{A}$ 从预言机获得的第一条密文。在实验中，$\mathcal{A}$ 接收一对密文 $(Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,0}))$（如果 $b = 0$），或一对密文 $(Enc_{pk}(m_{1,1}), Enc_{pk}(m_{2,1}))$（如果 $b = 1$）。我们用 $\mathcal{A}(pk, Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,0}))$ 表示 $\mathcal{A}$ 在第一种情况下的输出，第二种情况类似。

令 $\vec{C}_0$ 表示第一种情况下的密文对分布，$\vec{C}_1$ 表示第二种情况下的密文对分布。为了证明定义

11.5 成立（对于 $\mathrm{PubK}^{\mathrm{LR-cpa2}}_{\mathcal{A},\Pi}$），我们需要证明 $\mathcal{A}$ 无法区分是被给予一对服从 $\vec{C}_0$ 分布的密文，还是一对服从 $\vec{C}_1$ 分布的密文。也就是说，我们需要证明存在一个可忽略函数 negl，使得

$$|Pr[\mathcal{A}(pk, Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,0})) = 1] - Pr[\mathcal{A}(pk, Enc_{pk}(m_{1,1}), Enc_{pk}(m_{2,1})) = 1]| \leq$$

（这等同于定义 11.5，原因与定义 3.9 等同于定义 3.8 的原因相同。）为了证明这一点，我们将证明：

1. Π 的 CPA-安全性意味着 $\mathcal{A}$ 无法区分以下两种情况：被给予一对服从 $\vec{C}_0$ 分布的密文，或者一对密文 $(Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,1}))$，后者对应于加密 $\mathcal{A}$ 的第一次预言机查询中的第一条消息和第二次预言机查询中的第二条消息。（尽管这在 $\mathrm{PubK}^{\mathrm{LR-cpa2}}_{\mathcal{A},\Pi}(n)$ 中不会发生，我们仍然可以询问如果给定这样的密文对，$\mathcal{A}$ 的行为会是什么。）令 $\vec{C}_{01}$ 表示后一种情况下的密文对分布。
2. 类似地，Π 的 CPA-安全性意味着 $\mathcal{A}$ 无法区分被给予一对服从 $\vec{C}_{01}$ 分布的密文，还是一对服从 $\vec{C}_1$ 分布的密文。

上述说法表明 $\mathcal{A}$ 无法区分分布 $\vec{C}_0$ 和 $\vec{C}_{01}$，也无法区分分布 $\vec{C}_{01}$ 和 $\vec{C}_1$。我们得出结论（使用简单的代数），$\mathcal{A}$ 无法区分分布 $\vec{C}_0$ 和 $\vec{C}_1$。

那么，证明的关键在于证明 $\mathcal{A}$ 无法区分被给予一对服从 $\vec{C}_0$ 分布的密文，还是一对服从 $\vec{C}_{01}$ 分布的密文。（另一种情况类似地成立。）也就是说，我们想证明存在一个可忽略函数 negl，使得

$$|Pr[\mathcal{A}(pk, Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,0})) = 1] - Pr[\mathcal{A}(pk, Enc_{pk}(m_{1,0}), Enc_{pk}(m_{2,1})) = 1]| \leq$$

请注意，在这两种情况下，对手 $\mathcal{A}$ 的输入**仅在第二个元素上有所不同**。直觉上，不可区分性是从单消息情况推导出来的，因为 $\mathcal{A}$ 可以自己生成 $Enc_{pk}(m_{1,0})$。形式上，考虑在实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A}',\Pi}(n)$ 中运行的以下 PPT 对手 $\mathcal{A}'$：

对手 $\mathcal{A}'$：

1. 在输入 **pk** 时，对手 $\mathcal{A}'$ 将 $\mathcal{A}(pk)$ 作为子程序运行。
2. 当 $\mathcal{A}$ 对 LR 预言机进行第一次查询 $(m_{1,0}, m_{1,1})$ 时，$\mathcal{A}'$ 计算 $c_1 \leftarrow Enc_{pk}(m_{1,0})$ 并将 c_1 作为预言机的响应返回给 $\mathcal{A}$。
3. 当 $\mathcal{A}$ 对 LR 预言机进行第二次查询 $(m_{2,0}, m_{2,1})$ 时，$\mathcal{A}'$ 输出 $(m_{2,0}, m_{2,1})$ 并接收挑战密文 c_2。这作为 LR 预言机的响应返回给 $\mathcal{A}$。
4. $\mathcal{A}'$ 输出 $\mathcal{A}$ 输出的比特 b'。

查看实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A}',\Pi}(n)$，我们看到当 $b=0$ 时，挑战密文 c_2 被计算为 $Enc_{pk}(m_{2,0})$。因此，

$$Pr[\mathcal{A}'(Enc_{pk}(m_{2,0}))=1]=Pr[\mathcal{A}(Enc_{pk}(m_{1,0}),Enc_{pk}(m_{2,0}))=1]. \quad (11.4)$$

（我们省略了对 pk 的明确提及以节省空间。）相比之下，当实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A}',\Pi}(n)$ 中的 $b=1$ 时，c_2 被计算为 $Enc_{pk}(m_{2,1})$，因此

$$Pr[\mathcal{A}'(Enc_{pk}(m_{2,1}))=1]=Pr[\mathcal{A}(Enc_{pk}(m_{1,0}),Enc_{pk}(m_{2,1}))=1]. \quad (11.5)$$

Π 的 CPA-安全性意味着存在一个可忽略函数 negl，使得

$$|Pr[\mathcal{A}'(Enc_{pk}(m_{2,0}))=1]-Pr[\mathcal{A}'(Enc_{pk}(m_{2,1}))=1]|\leq \mathrm{negl}(n).$$

这与等式 (11.4) 和 (11.5) 一起得出等式 (11.3)。以**几乎完全相同**的方式，我们可以证明：

$$|Pr[\mathcal{A}(pk,Enc_{pk}(m_{1,0}),Enc_{pk}(m_{2,1}))=1]-Pr[\mathcal{A}(pk,Enc_{pk}(m_{1,1}),Enc_{pk}(m_{2,1}))=1]|\leq$$

等式 (11.2) 通过组合等式 (11.3) 和 (11.6) 得出。在一般情况下出现的主要复杂性是，对 LR 预言机的查询次数不再固定，而是可能是 n 的任意**多项式**。在形式证明中，这是使用**混合论证**（hybrid argument）来处理的。（混合论证也用于第 7 章。）

证明（定理 11.6） 令 Π 是一个 CPA-安全的公钥加密方案，$\mathcal{A}$ 是实验 $\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n)$ 中的任意 PPT 对手。令 $t=t(n)$ 是 $\mathcal{A}$ 对 LR 预言机进行的查询次数的多项式上限，并且假设不失一般性，$\mathcal{A}$ 总是正好查询预言机这么多次。对于给定的公钥 **pk** 和 $0\leq i\leq t$，令 LR^i_{pk} 表示这样一个预言机，它在输入 (m_0,m_1) 时，对于它接收到的**前 i 次**查询返回 $Enc_{pk}(m_0)$，并对于它接收到的**剩余 $t-i$ 次**查询返回 $Enc_{pk}(m_1)$。（也就是说，对于前 i 次查询，加密输入对中的第一条消息，并作为对剩余查询的响应，加密输入对中的第二条消息。）我们强调，每次加密都使用**均匀、独立的随机性**来计算。

使用此符号，我们有

$$Pr[\mathrm{PubK}^{\mathrm{LR-cpa}}_{\mathcal{A},\Pi}(n)=1]=\frac{1}{2}\cdot Pr[\mathcal{A}^{LR^t_{pk}}(pk)=0]+\frac{1}{2}\cdot Pr[\mathcal{A}^{LR^0_{pk}}(pk)=1]$$

因为，从 $\mathcal{A}$（它正好进行 t 次查询）的角度来看，预言机 LR^t_{pk} 等同于 $LR_{pk,0}$，预言机 LR^0_{pk} 等同于 $LR_{pk,1}$。为了证明 Π 满足定义 11.5，我们将证明对于任何 PPT $\mathcal{A}$，存在一个可忽略函数 negl'，使得

$$|Pr[\mathcal{A}^{LR^t_{pk}}(pk)=1]-Pr[\mathcal{A}^{LR^0_{pk}}(pk)=1]|\leq \mathrm{negl}'(n). \quad (11.7)$$

（与以前一样，这等同于定义 11.5，原因与定义 3.9 等同于定义 3.8 的原因相同。）

考虑以下窃听单个消息加密的 PPT 对手 $\mathcal{A}'$：

对手 $\mathcal{A}'$：

1. $\mathcal{A}'$，给定 **pk**，选择一个均匀索引 $i \leftarrow \{1, \ldots, t\}$。
2. $\mathcal{A}'$ 运行 $\mathcal{A}(pk)$，并如下回答其第 j 次预言机查询 $(m_{j,0}, m_{j,1})$：(a) 对于 $j < i$，对手 $\mathcal{A}'$ 计算 $c_j \leftarrow Enc_{pk}(m_{j,0})$ 并将 c_j 作为其预言机的响应返回给 $\mathcal{A}$。(b) 对于 $j = i$，对手 $\mathcal{A}'$ 输出 $(m_{j,0}, m_{j,1})$ 并接收挑战密文 c_j。这作为其预言机的响应返回给 $\mathcal{A}$。(c) 对于 $j > i$，对手 $\mathcal{A}'$ 计算 $c_j \leftarrow Enc_{pk}(m_{j,1})$ 并将 c_j 作为其预言机的响应返回给 $\mathcal{A}$。
3. $\mathcal{A}'$ 输出 $\mathcal{A}$ 输出的比特 b'。

考虑实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A}',\Pi}(n)$。固定某个选择 $i = i^*$，请注意，如果 c_{i^*} 是 $m_{i^*,0}$ 的加密，则 $\mathcal{A}$ 与其预言机的交互与与预言机 $LR^{i^*}_{pk}$ 的交互**相同**。因此，

$$Pr[\mathcal{A}' \text{ outputs } 1|b=0] = \sum_{i^*=1}^{t} Pr[i=i^*] \cdot Pr[\mathcal{A}' \text{ outputs } 1|b=0 \wedge i=i^*] = \sum_{i^*=1}^{t} \frac{1}{t} \cdot Pr[\mathcal{A}^{LR}$$

另一方面，如果 c_{i^*} 是 $m_{i^*,1}$ 的加密，则 $\mathcal{A}$ 与其预言机的交互与与预言机 $LR^{i^*-1}_{pk}$ 的交互**相同**，因此

$$Pr[\mathcal{A}' \text{ outputs } 1|b=1] = \sum_{i^*=1}^{t} Pr[i=i^*] \cdot Pr[\mathcal{A}' \text{ outputs } 1|b=1 \wedge i=i^*] = \sum_{i^*=1}^{t} \frac{1}{t} \cdot Pr[\mathcal{A}^{LR}$$

其中第三个等式是通过简单地移动求和索引获得的。

由于 $\mathcal{A}'$ 在多项式时间内运行，因此 Π 是 CPA-安全的假设意味着存在一个可忽略函数 negl，使得

$$|Pr[\mathcal{A}' \text{ outputs } 1|b=0] - Pr[\mathcal{A}' \text{ outputs } 1|b=1]| \leq \mathrm{negl}(n).$$

但这又意味着

$$\mathrm{negl}(n) \geq |\sum_{i^*=1}^{t} \frac{1}{t} \cdot Pr[\mathcal{A}^{LR^{i^*}_{pk}}(pk)=1] - \sum_{i^*=0}^{t-1} \frac{1}{t} \cdot Pr[\mathcal{A}^{LR^{i^*}_{pk}}(pk)=1]| = \frac{1}{t} \cdot |Pr[\mathcal{A}^{LR^{t}_{pk}}(pk) =$$

因为两次求和中除了一个项以外的所有项都相互抵消了。我们得出结论

$$|Pr[\mathcal{A}^{LR^{t}_{pk}}(pk)=1]-Pr[\mathcal{A}^{LR^{0}_{pk}}(pk)=1]|\leq t(n)\cdot \mathrm{negl}(n).$$

因为 t 是多项式，所以函数 $t\cdot \mathrm{negl}(n)$ 是可忽略的。由于 $\mathcal{A}$ 是任意的 PPT 对手，这表明等式 (11.7) 成立，从而完成了 Π 具有不可区分多次加密的证明。

11.2.3 抵抗选择密文攻击的安全性

选择密文攻击（chosen-ciphertext attacks，其中对手能够获得其选择的任意密文的解密——但有一个技术限制，如下所述）在公钥设置中是一个问题，就像在私钥设置中一样。事实上，在公钥设置中，它们可以说是**更**令人担忧的问题，因为在公钥设置中，接收方期望接收来自**多个事先可能未知**的发送方的密文，而在私钥设置中，接收方打算仅使用任何特定的秘密密钥与**单个已知**的发送方进行通信。

假设窃听者 $\mathcal{A}$ 观察到发送方 S 发送给接收方 R 的密文 c。广义上讲，在公钥设置中有两类选择密文攻击：

- $\mathcal{A}$ 可能代表 S 向 R 发送修改后的密文 c'。（例如，在加密电子邮件的背景下，$\mathcal{A}$ 可能会构造一封加密电子邮件 c' 并伪造“发件人”字段，使其看起来像是源自 S。）在这种情况下，尽管 $\mathcal{A}$ 不太可能获得 c' 的**完整解密** m'，但 $\mathcal{A}$ 可能会根据 R 的后续行为**推断出**有关 m' 的一些信息。基于此信息，$\mathcal{A}$ 可能能够了解有关原始消息 m 的某些信息。
- $\mathcal{A}$ 可能会以**自己的名义**向 R 发送修改后的密文 c'。在这种情况下，如果 R 直接回复 $\mathcal{A}$，$\mathcal{A}$ 可能会获得 c' 的**完整解密** m'。即使 $\mathcal{A}$ 对 m' 一无所知，这个修改后的消息可能与原始消息 m 存在某种**已知关系**，$\mathcal{A}$ 可以利用这种关系；请参阅下面的第三种情况以获取示例。

第二类攻击是公钥加密特有的，在私钥设置中没有类似物。

不难确定一些说明上述攻击类型的现实场景：

场景 1。 假设用户 S 通过向其银行发送加密的密码 pw 与时间戳的串联来登录其银行账户。进一步假设银行发送两种类型的错误消息：如果加密的密码与 S 存储的密码不匹配，则返回“密码不正确”；如果密码正确但时间戳不正确，则返回“时间戳不正确”。如果对手获得 S 发送给银行的密文 c，对手现在可以代表 S 向银行发送密文 c' 并观察产生的错误消息，从而发起选择密文攻击。（这类似于我们在 3.7.2 节中看到的**填充预言机攻击**。）在某些情况下，这些信息可能足以让对手确定用户的**整个密码**。

场景 2。 假设 S 向 R 发送一封加密电子邮件 c，$\mathcal{A}$ 观察到这封电子邮件。如果 $\mathcal{A}$ 以自己的名义向 R 发送一封加密电子邮件 c'，则 R 可能会回复此电子邮件并引用与 c' 对应的解密文本

m'。在这种情况下，R 实际上充当了 $\mathcal{A}$ 的**解密预言机**，并可能解密 $\mathcal{A}$ 发送的任何密文。

场景 3。 与选择密文安全性密切相关的一个问题是密文的**潜在可塑性**（potential malleability）。由于正式定义相当复杂，我们在此不提供，而是只给出直观的想法。如果一个方案具有以下属性，则它是**可塑的**（malleable）：给定某个未知消息 m 的加密 c，可以想出密文 c'，它是消息 m' 的加密，其中 m' 与 m 存在某种**已知**的关系。例如，也许给定 m 的加密，可以构造 $2m$ 的加密。（我们稍后将看到具有此属性和类似属性的 CPA-安全方案的自然示例；参见 13.2.3 节。）

现在想象 R 正在运行一个拍卖，其中 S 和 $\mathcal{A}$ 两方通过使用 R 的公钥加密来提交他们的出价。如果使用可塑加密方案，则对手 $\mathcal{A}$ 可能总是能够（在不出最高价的情况下）以最高价出价，方法是执行以下攻击：等待 S 发送对应于其出价 m（$\mathcal{A}$ 未知）的密文 c；然后发送对应于出价 $m' = 2m$ 的密文 c'。请注意，在 R 宣布结果之前，m（以及 m'）对于 $\mathcal{A}$ 仍然是未知的，因此发生此类攻击的可能性**并不矛盾**于加密方案是 CPA-安全的这一事实。另一方面，**CCA-安全**的方案可以被证明是**不可塑的**（non-malleable），这意味着它们不容易受到此类攻击。

定义。 抵抗选择密文攻击的安全性是通过对私钥设置中的类似定义（定义 3.33）进行适当修改来定义的。给定一个公钥加密方案 Π 和一个对手 $\mathcal{A}$，考虑以下实验：

CCA 不可区分性实验 $\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 **(pk, sk)**。
2. 给予对手 $\mathcal{A}$ **pk** 和对**解密预言机** $Dec_{sk}(\cdot)$ 的访问权限。它输出一对**等长**的消息 m_0, m_1。（这些消息必须在与 **pk** 相关的消息空间中。）
3. 选择一个均匀的比特 $b \in \{0,1\}$，然后计算密文 $c \leftarrow Enc_{pk}(m_b)$ 并将其给予 $\mathcal{A}$。
4. $\mathcal{A}$ 继续与解密预言机交互，但**不得请求**解密 c 本身。最后，$\mathcal{A}$ 输出一个比特 b'。
5. 如果 $b' = b$，则实验的输出定义为 1，否则为 0。

定义 11.8 如果对于所有概率多项式时间对手 $\mathcal{A}$，存在一个可忽略函数 negl，使得

$$Pr[\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$$

则公钥加密方案 $\Pi = (Gen, Enc, Dec)$ 在**选择密文攻击**下具有**不可区分加密**（或称为 **CCA-安全**）。

定理 11.6 的自然类比也适用于 CCA-安全性。也就是说，如果一个方案在选择密文攻击下具有不可区分加密，那么它在选择密文攻击下也具有不可区分多次加密（其中后者被适当定义）。然

而，有趣的是，主张 11.7 的类比**不**适用于 CCA-安全性。

与定义 3.33 中一样，我们必须阻止攻击者将挑战密文 c 提交给解密预言机，以使该定义可实现。但这种限制并没有使定义变得毫无意义，特别是对于前面给出的三种激发性场景中的每一种，都可以论证设置 $c' = c$ 对攻击者没有益处：

- 在涉及基于密码登录的第一种场景中，攻击者通过向银行重放 c 来了解 S 的密码一无所知，因为在这种情况下，攻击者已经知道不会产生错误消息。
- 在涉及加密电子邮件的第二种场景中，向接收方发送 $c' = c$ 很可能会使接收方生疑，因此它会完全拒绝回复。
- 在涉及拍卖的最后一种场景中，如果对手的加密出价与另一方的加密出价相同，则 R 可以很容易地检测到作弊。无论如何，在这种情况下，攻击者通过重放 c 所实现的只是它提交了与诚实方相同的出价。

11.3 混合加密和 KEM/DEM 范式

主张 11.7 表明任何用于 l'-比特消息的 CPA-安全公钥加密方案都可以用于获得用于**任意长度消息**的 CPA-安全公钥加密方案。使用这种方法加密一条 l-比特消息需要 $\gamma \triangleq \lceil l/l' \rceil$ 次调用原始加密方案，这意味着计算和密文长度相对于底层方案都会增加 γ 的乘法因子。

通过将私钥加密与公钥加密结合使用，可以做得更好。这提高了效率，因为**私钥加密比公钥加密快得多**，并且提高了带宽，因为**私钥方案具有更低的密文扩展**。由此产生的组合称为**混合加密**（hybrid encryption），并在实践中得到广泛使用。基本思想是**使用公钥加密来获得一个共享密钥 k**，然后**使用私钥加密方案和密钥 k 来加密消息 m**。接收方使用其长期（非对称）私钥来派生 k，然后使用私钥解密（使用密钥 k）来恢复原始消息。我们强调，尽管使用了私钥加密作为组件，但这仍然是一个**成熟的公钥加密方案**，因为发送方和接收方事先不共享任何秘密密钥。

在一个直接实现此想法的方案中（见图 11.1），发送方将通过 (1) **选择一个均匀值 k**，然后 (2) **使用公钥加密方案加密 k** 来共享 k。一种更直接的方法是使用一种称为**密钥封装机制 (KEM)** 的公钥原语来“一举”完成这两件事。正如我们稍后将看到的，这在概念上和效率上都具有优势。

一个 KEM 具有三个算法，其精神类似于公钥加密方案的算法。与以前一样，密钥生成算法 **Gen** 用于生成一对公钥和私钥。取代加密，我们现在有一个**封装算法 Encaps**，它**只**以公钥作为输入（**没有**消息），并输出一个密文 c 和一个密钥 k。接收方运行相应的**解封装算法 Decaps**，使用私钥从密文 c 中恢复 k。形式上：

定义 11.9 密钥封装机制 (KEM) 是一个由概率多项式时间算法组成的**四元组 (Gen,**

Encaps, Decaps)，满足以下条件：

1. **密钥生成算法 Gen** 以安全参数 1^n 为输入，并输出一对公钥/私钥 **(pk, sk)**。我们假设 **pk** 和 **sk** 的长度都至少为 n，并且可以从 **pk** 确定 n。
2. **封装算法 Encaps** 以公钥 **pk** 和安全参数 1^n 为输入。它输出一个密文 c 和一个密钥 $k \in \{0,1\}^{l(n)}$，其中 l 是密钥长度。我们记为 $(c,k) \leftarrow Encaps_{pk}(1^n)$。
3. **确定性解封装算法 Decaps** 以私钥 **sk** 和密文 c 为输入，输出一个密钥 k 或表示失败的特殊符号 $\perp$。我们记为 $k := Decaps_{sk}(c)$。

要求除了由 $Gen(1^n)$ 输出的 **(sk, pk)** 的可忽略概率外，如果 $Encaps_{pk}(1^n)$ 输出 (c,k)，则 $Decaps_{sk}(c)$ 输出 k。

在定义中，为简单起见，我们假设 Encaps 总是输出（一个密文 c 和）一个固定长度 $l(n)$ 的密钥。也可以考虑一个更一般的定义，其中 Encaps 将 1^l 作为附加输入，并输出长度为 l 的密钥。

任何公钥加密方案都可以通过选择一个随机密钥 k 并对其进行加密，从而**平凡地**给出一个 KEM。然而，正如我们将看到的，专门的 KEM 构造可以更高效。

使用 KEM（密钥长度为 n），我们可以实现如图 11.2 所示的混合加密。发送方运行 $Encaps_{pk}(1^n)$ 以获得 c 和密钥 k；然后它使用私钥加密方案加密其消息 m，使用 k 作为密钥。在这种情况下，私钥加密方案称为**数据封装机制 (DEM)**，原因很明显。发送给接收方的密文包括 c 和来自私钥方案的密文 c'。构造 11.10 给出了形式规范。

由此产生的混合加密方案 Π^{hy} 的效率如何？对于某个固定的 n 值，令 α 表示使用 Encaps 封装 n-比特密钥的成本，令 β 表示使用 Enc' 加密的成本（每位明文）。假设 $|m| > n$，这是有趣的情况。那么，使用 Π^{hy} 加密消息 m 的成本（每位明文）为

$$\frac{\alpha + \beta \cdot |m|}{|m|} = \frac{\alpha}{|m|} + \beta, \quad (11.8)$$

对于足够长的 m，这接近于 β。因此，在消息非常长的情况下，公钥加密方案 Π^{hy} 产生的每位成本与私钥方案 Π' 的每位成本**相同**。因此，混合加密允许我们**以私钥加密的效率实现公钥加密的功能**，至少对于足够长的消息是如此。

类似的计算可用于衡量混合加密对密文长度的影响。对于某个固定的 n 值，令 L 表示 Encaps 输出的密文长度，并假设使用 Enc' 对消息 m 的私钥加密产生的密文长度为 $n + |m|$（这可以使用 3.6 节中讨论的一种加密模式实现；实际上，甚至**密文长度** $|m|$ 也是可能的，因为正如我们将看到的，Π' 不一定需要是 CPA-安全的）。那么，方案 Π^{hy} 中的密文总长度为

$$L + n + |m|. \quad (11.9)$$

相比之下，当使用如等式 (11.1) 中的逐块加密时，并假设使用 Enc 对 n-比特消息的公钥加密产生的密文长度为 L，则对消息 m 的加密将产生长度为 $L \cdot \lceil |m|/n \rceil$ 的密文。等式 (11.9) 中报告的密文长度对于足够长的 m 是一个**显着改进**。

我们可以使用一些粗略的估计来了解上述结果在实践中意味着什么。（我们强调，这些数字仅旨在让读者对改进有所了解；实际值将取决于各种因素。）密钥 k 长度的典型值可能是 $n = 128$。此外，“原生”公钥加密方案在加密 128 位消息时可能会产生 256 位密文；假设 KEM 在封装 128 位密钥时具有相同的密文长度。令 α 像以前一样表示 128 位密钥的公钥加密/封装的计算成本，我们看到如等式 (11.1) 中的逐块加密将加密 1 MB $(= 10^6\text{-bit})$ 消息，计算成本约为 $\alpha \cdot \lceil 10^6/128 \rceil \approx 7800 \cdot \alpha$，密文长度为 2 MB。将其与混合加密的效率进行比较。令 β 像以前一样表示私钥加密的每位计算成本，一个合理的近似是 $\beta \approx \alpha/10^5$。使用等式 (11.8)，我们看到对于 1 Mb 消息的混合加密的总计算成本为

$$\alpha + 10^6 \cdot \frac{\alpha}{10^5} = 11 \cdot \alpha,$$

密文将仅略长于 1 MB。因此，在这种情况下，混合加密将计算效率提高了 700 倍，密文长度提高了 2 倍。

仍然需要分析 Π^{hy} 的安全性。这当然取决于其底层组件 Π 和 Π' 的安全性。在接下来的部分中，我们定义了 KEM 的 CPA-安全性和 CCA-安全性的概念，并表明：

- 如果 Π 是一个 CPA-安全 KEM 并且私钥方案 Π' 在窃听者存在下具有不可区分加密，则构造 11.10 中的 Π^{hy} 是一个 **CPA-安全公钥加密方案**。请注意，Π' **只需**满足一个**较弱**的安全性定义，回想一下，这在私钥设置中**并不蕴含 CPA-安全性**，即可使混合方案 Π^{hy} 成为 CPA-安全的。直觉上，原因是**每次**加密新消息时都会选择一个**新的、均匀的密钥** k。由于每个密钥 k 只使用**一次**，因此 Π' 的单个加密的不可区分性足以保证混合方案 Π^{hy} 的安全性。这意味着使用**伪随机生成器**（或流密码）进行基本私钥加密（如构造 3.17 中）就足够了。
- 如果 Π 是一个 CCA-安全 KEM 并且 Π' 是一个 CCA-安全私钥加密方案，则 Π^{hy} 是一个 **CCA-安全公钥加密方案**。

构造 11.10

令 $\Pi = (Gen, Encaps, Decaps)$ 是一个密钥长度为 n 的 KEM，令 $\Pi' = (Gen', Enc', Dec')$ 是一个私钥加密方案。按如下方式构造公钥加密方案 $\Pi^{\text{hy}} = (Gen^{\text{hy}}, Enc^{\text{hy}}, Dec^{\text{hy}})$：

- Gen^{hy}：在输入 1^n 时，运行 $Gen(1^n)$ 并使用输出的公钥和私钥 **(pk, sk)**。
- Enc^{hy}：在输入公钥 **pk** 和消息 $m \in \{0,1\}^*$ 时：
  i. 计算 $(c, k) \leftarrow Encaps_{pk}(1^n)$。
  ii. 计算 $c' \leftarrow Enc'_k(m)$。
  iii. 输出密文 $\langle c, c' \rangle$。
- Dec^{hy}：在输入私钥 **sk** 和密文 $\langle c, c' \rangle$ 时：
  i. 计算 $k := Decaps_{sk}(c)$。
  ii. 输出消息 $m := Dec'_k(c')$。

使用 KEM/DEM 范式的混合加密。

11.3.1 CPA-安全性

为简单起见，在本节和下一节中，我们假设 KEM 的密钥长度为 n。我们通过类比定义 11.2 来定义 KEM 的 CPA-安全性的概念。与那里一样，这里的对手窃听**单个密文** c。定义 11.2 要求攻击者无法区分 c 是某个消息 m_0 的加密还是某个其他消息 m_1 的加密。对于 KEM，**没有**消息，我们转而要求**封装的密钥 k 与独立于密文 c 的均匀密钥不可区分**。

令 $\Pi = (Gen, Encaps, Decaps)$ 是一个 KEM，$\mathcal{A}$ 是一个任意对手。

CPA 不可区分性实验 $\mathrm{KEM}^{\text{cpa}}_{\mathcal{A},\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 **(pk, sk)**。然后运行 $Encaps_{pk}(1^n)$ 以生成 (c, k)，其中 $k \in \{0,1\}^n$。
2. 选择一个均匀的比特 $b \in \{0,1\}$。如果 $b = 0$，设置 $\hat{k} := k$。如果 $b = 1$，则选择一个均匀的 $\hat{k} \in \{0,1\}^n$。
3. 将 $(pk, c, \hat{k})$ 给予 $\mathcal{A}$，$\mathcal{A}$ 输出一个比特 b'。如果 $b' = b$，则实验的输出定义为 1，否则为 0。

在实验中，$\mathcal{A}$ 被给予密文 c 以及**要么**与 c 对应的实际密钥 k，**要么**一个独立的均匀密钥。如果任何高效对手都无法区分这两种可能性，则 KEM 是 CPA-安全的。

定义 11.11 如果对于所有概率多项式时间对手 $\mathcal{A}$，存在一个可忽略函数 negl，使得

$$Pr[\mathrm{KEM}^{\text{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

则密钥封装机制 Π 是 **CPA-安全**的。

在本节的其余部分，我们证明以下定理：

定理 11.12 如果 Π 是一个 CPA-安全 KEM 并且 Π' 是一个在窃听者存在下具有不可区分加密的私钥加密方案，则如构造 11.10 中的 Π^{hy} 是一个 **CPA-安全公钥加密方案**。

在形式化证明定理之前，我们给出一些直觉。令符号 "$X \equiv Y$" 表示**任何**多项式时间对手都**无法区分**两个分布 X 和 Y 之间的差异。（这个概念在 7.8 节中得到了更正式的处理，尽管我们在这里不依赖该节。）例如，令 $Encaps_{pk}^{(1)}(1^n)$（分别 $Encaps_{pk}^{(2)}(1^n)$）表示 Encaps 输出的密文（分别密钥）。Π 是 CPA-安全的这一事实意味着

$$(pk, Encaps_{pk}^{(1)}(1^n), Encaps_{pk}^{(2)}(1^n)) \equiv (pk, Encaps_{pk}^{(1)}(1^n), k'),$$

其中 pk 由 $Gen(1^n)$ 生成，k' 是独立且均匀地从 $\{0,1\}^n$ 中选择的。类似地，Π' 在窃听者存在下具有不可区分加密的这一事实意味着，对于 $\mathcal{A}$ 输出的任何 m_0, m_1，如果 k 是均匀随机选择的，我们有 $Enc'_k(m_0) \equiv Enc'_k(m_1)$。

为了证明 Π^{hy} 的 CPA-安全性，我们需要证明

$$(pk, Encaps_{pk}^{(1)}(1^n), Enc'_k(m_0)) \equiv (pk, Encaps_{pk}^{(1)}(1^n), Enc'_k(m_1)) \quad (11.10)$$

对于 PPT 对手 $\mathcal{A}$ 输出的 m_0, m_1 成立。（等式 (11.10) 足以证明 Π^{hy} 在窃听者存在下具有不可区分加密，根据命题 11.3，这蕴含 Π^{hy} 是 CPA-安全的。）

证明分三步进行。（见图 11.3。）首先我们证明

$$(pk, Encaps_{pk}^{(1)}(1^n), Enc'_k(m_0)) \equiv (pk, Encaps_{pk}^{(1)}(1^n), Enc'_{k'}(m_0)), \quad (11.11)$$

其中左侧的 k 由 $Encaps_{pk}^{(2)}(1^n)$ 输出，右侧的 k' 是一个独立的均匀密钥。这通过一个相当直接的归约得出，因为 Π 的 CPA-安全性恰好意味着 $Encaps_{pk}^{(2)}(1^n)$ 即使在给定 pk 和 $Encaps_{pk}^{(1)}(1^n)$ 的情况下，也无法与均匀密钥 k' 区分。

接下来，我们证明

$$(pk, Encaps_{pk}^{(1)}(1^n), Enc'_{k'}(m_0)) \equiv (pk, Encaps_{pk}^{(1)}(1^n), Enc'_{k'}(m_1)). \quad (11.12)$$

这里的区别在于使用 Π' 和均匀、独立的密钥 k' 加密 m_0 还是 m_1。因此，等式 (11.12) 使用 Π' 在窃听者存在下具有不可区分加密的事实得出。

正如等式 (11.11) 的情况一样，我们也可以证明

$$(pk, Encaps_{pk}^{(1)}(1^n), Enc'_k(m_1)) \equiv (pk, Encaps_{pk}^{(1)}(1^n), Enc'_{k'}(m_1)), \quad (11.13)$$

再次依赖于 Π 的 CPA-安全性。等式 (11.11)-(11.13) 通过**传递性**蕴含等式 (11.10) 这个期望的结果。（传递性将在我们下面给出的证明中隐含。）

我们现在给出完整的证明。

证明（定理 11.12） 我们证明 Π^{hy} 在窃听者存在下具有不可区分加密；根据命题 11.3，这蕴含它是 CPA-安全的。

固定一个任意的 PPT 对手 $\mathcal{A}^{\text{hy}}$，并考虑实验 $\text{PubK}^{\text{eav}}_{\mathcal{A}^{\text{hy}},\Pi^{\text{hy}}}(n)$。我们的目标是证明存在一个可忽略函数 negl，使得

$$Pr[\text{PubK}^{\text{eav}}_{\mathcal{A}^{\text{hy}},\Pi^{\text{hy}}}(n)=1] \leq \frac{1}{2}+\text{negl}(n).$$

根据实验的定义，我们有

$$Pr[\text{PubK}^{\text{eav}}_{\mathcal{A}^{\text{hy}},\Pi^{\text{hy}}}(n)=1] = \frac{1}{2}\cdot Pr[\mathcal{A}^{\text{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_k(m_0))=0] + \frac{1}{2}\cdot Pr[\mathcal{A}^{\text{hy}}(pk$$

其中在每种情况下 k 等于 $Encaps^{(2)}_{pk}(1^n)$。考虑攻击 Π 的以下 PPT 对手 $\mathcal{A}_1$。

对手 $\mathcal{A}_1$：

1. $\mathcal{A}_1$ 被给予 $(pk, c, \hat{k})$。
2. $\mathcal{A}_1$ 运行 $\mathcal{A}^{\text{hy}}(pk)$ 以获得两条消息 m_0, m_1。然后 $\mathcal{A}_1$ 计算 $c' \leftarrow Enc'_{\hat{k}}(m_0)$，将密文 $\langle c, c'\rangle$ 给予 $\mathcal{A}^{\text{hy}}$，并输出 $\mathcal{A}^{\text{hy}}$ 输出的比特 b'。

考虑 $\mathcal{A}_1$ 在实验 $\text{KEM}^{\text{cpa}}_{\mathcal{A}_1,\Pi}(n)$ 中攻击 Π 的行为。当该实验中的 $b=0$ 时，$\mathcal{A}_1$ 被给予 $(pk, c, \hat{k})$，其中 c 和 $\hat{k}$ 都是由 $Encaps_{pk}(1^n)$ 输出的。这意味着 $\mathcal{A}^{\text{hy}}$ 被给予 $\langle c, c'\rangle = \langle c, Enc'_k(m_0)\rangle$ 形式的密文，其中 k 是由 c 封装的密钥。因此，

$$Pr[\mathcal{A}_1 \text{ outputs } 0 | b=0] = Pr[\mathcal{A}^{\text{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_k(m_0))=0].$$

另一方面，当实验 $\text{KEM}^{\text{cpa}}_{\mathcal{A}_1,\Pi}(n)$ 中的 $b=1$ 时，$\mathcal{A}_1$ 被给予 $(pk, c, \hat{k})$，其中 $\hat{k}$ 是均匀的并且独立于 c。如果我们将这样的密钥表示为 k'，这意味着 $\mathcal{A}^{\text{hy}}$ 被给予 $\langle c, Enc'_{k'}(m_0)\rangle$ 形式的密文，并且

$$Pr[\mathcal{A}_1 \text{ outputs } 1 | b=1] = Pr[\mathcal{A}^{\text{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m_0))=1].$$

由于 Π 是一个 CPA-安全 KEM，存在一个可忽略函数 negl_1 使得

$$\frac{1}{2} + \mathrm{negl}_1(n) \geq Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A}_1,\Pi}(n) = 1] = \frac{1}{2} \cdot Pr[\mathcal{A}_1 \text{ outputs } 0 | b = 0] + \frac{1}{2} \cdot Pr[\mathcal{A}_1 \text{ outputs } 1$$

$$= \frac{1}{2} \cdot Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_k(m_0)) = 0] + \frac{1}{2} \cdot Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m$$

其中 k 等于 $Encaps^{(2)}_{pk}(1^n)$ 并且 k' 是一个均匀且独立的密钥。

接下来，考虑窃听使用私钥方案 Π' 加密的消息的以下 PPT 对手 $\mathcal{A}'$。

对手 $\mathcal{A}'$：

1. $\mathcal{A}'(1^n)$ 自己运行 $Gen(1^n)$ 以生成密钥 **(pk, sk)**。它也计算 $c \leftarrow Encaps^{(1)}_{pk}(1^n)$。
2. $\mathcal{A}'$ 运行 $\mathcal{A}^{\mathrm{hy}}(pk)$ 以获得两条消息 m_0, m_1。 $\mathcal{A}'$ 输出它们，并返回密文 c'。
3. $\mathcal{A}'$ 将密文 $\langle c, c' \rangle$ 给予 $\mathcal{A}^{\mathrm{hy}}$ 并输出 $\mathcal{A}^{\mathrm{hy}}$ 输出的比特 b'。

当实验 $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A}',\Pi'}(n)$ 中的 $b = 0$ 时，对手 $\mathcal{A}'$ 被给予密文 c'，它是使用均匀且独立于其他任何事物的密钥 k' 对 m_0 的加密。因此，$\mathcal{A}^{\mathrm{hy}}$ 被给予 $\langle c, Enc'_{k'}(m_0) \rangle$ 形式的密文，其中 k' 均匀且独立于 c，并且

$$Pr[\mathcal{A}' \text{ outputs } 0 | b = 0] = Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m_0)) = 0].$$

另一方面，当实验 $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A}',\Pi'}(n)$ 中的 $b = 1$ 时，$\mathcal{A}'$ 被给予使用均匀、独立密钥 k' 对 m_1 的加密。这意味着 $\mathcal{A}^{\mathrm{hy}}$ 被给予 $\langle c, Enc'_{k'}(m_1) \rangle$ 形式的密文，因此

$$Pr[\mathcal{A}' \text{ outputs } 1 | b = 1] = Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m_1)) = 1].$$

由于 Π' 在窃听者存在下具有不可区分加密，存在一个可忽略函数 negl' 使得

$$\frac{1}{2} + \mathrm{negl}'(n) \geq Pr[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A}',\Pi'}(n) = 1] \quad (11.16)$$

$$= \frac{1}{2} \cdot Pr[\mathcal{A}' \text{ outputs } 0 | b = 0] + \frac{1}{2} \cdot Pr[\mathcal{A}' \text{ outputs } 1 | b = 1]$$

$$= \frac{1}{2} \cdot Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m_0)) = 0] + \frac{1}{2} \cdot Pr[\mathcal{A}^{\mathrm{hy}}(pk, Encaps^{(1)}_{pk}(1^n), Enc'_{k'}(m$$

继续我们证明等式 (11.15) 的方式，我们可以证明存在一个可忽略函数 negl_2 使得

$$\frac{1}{2} + \mathrm{negl}_2(n) \geq Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A}_2,\Pi}(n) = 1] \quad (11.17)$$

$$= \frac{1}{2} \cdot Pr[\mathcal{A}_2 \text{ outputs } 0 | b = 0] + \frac{1}{2} \cdot Pr[\mathcal{A}_2 \text{ outputs } 1 | b = 1]$$

$$= \frac{1}{2} \cdot Pr[\mathcal{A}^{\text{hy}}(pk, Encaps_{pk}^{(1)}(1^n), Enc'_k(m_1)) = 1] + \frac{1}{2} \cdot Pr[\mathcal{A}^{\text{hy}}(pk, Encaps_{pk}^{(1)}(1^n), Enc'_{k'}(m$$

对等式 (11.15)-(11.17) 求和，并使用三个可忽略函数的和是可忽略的事实，我们看到存在一个可忽略函数 negl 使得

$$\begin{aligned}\frac{3}{2} + \text{negl}(n) \geq \frac{1}{2} \cdot (&Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_k(m_0)) = 0] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_0)) = 1] \\ &+ Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_0)) = 0] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_1)) = 1] \\ &+ Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_k(m_1)) = 1] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_1)) = 0]),\end{aligned}$$

其中在上述所有情况中 $c = Encaps_{pk}^{(1)}(1^n)$。请注意

$$Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_0)) = 1] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_0)) = 0] = 1,$$

因为互补事件的概率总是和为 1。类似地，

$$Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_1)) = 1] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_{k'}(m_1)) = 0] = 1.$$

因此，

$$\frac{1}{2} + \text{negl}(n) \geq \frac{1}{2} \cdot (Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_k(m_0)) = 0] + Pr[\mathcal{A}^{\text{hy}}(pk, c, Enc'_k(m_1)) = 1]) = Pr[$$

（使用等式 (11.14) 作为最后一个等式），从而证明了该定理。

11.3.2 CCA-安全性

如果私钥加密方案 Π' 本身不安全地抵抗选择密文攻击，那么（无论使用哪种 KEM）由此产生的混合加密方案 Π^{hy} 也不是 CCA-安全的。作为一个简单的说明性示例，假设我们选择构造 3.17 作为我们的私钥加密方案。那么，在 KEM 未指定的情况下，Π^{hy} 对消息 m 的加密是通过计算 $(c, k) \leftarrow Encaps_{pk}(1^n)$ 然后输出密文 $\langle c, G(k) \oplus m \rangle$，其中 G 是一个伪随机生成器。给定密文 $\langle c, c' \rangle$，攻击者可以简单地**翻转** c' 的最后一位，以获得一个修改后的密文，它是 m（最后一位被翻转）的有效加密。

解决这个问题的自然方法是使用 **CCA-安全**的私钥加密方案。但这显然不够，如果 KEM 容易受到选择密文攻击。由于我们尚未定义此概念，我们现在就来定义。

与定义 11.11 中一样，我们要求给定密文 c 的对手**无法区分**该密文封装的密钥 k 与均匀且独立

的密钥 k'。然而，现在我们**额外**允许攻击者请求解封装其选择的密文（只要它们不同于挑战密文）。形式上，令 $\Pi = (Gen, Encaps, Decaps)$ 是一个密钥长度为 n 的 KEM，$\mathcal{A}$ 是一个对手，并考虑以下实验：

CCA 不可区分性实验 $\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 **(pk, sk)**。然后运行 $Encaps_{pk}(1^n)$ 以生成 (c, k)，其中 $k \in \{0,1\}^n$。
2. 选择一个均匀的比特 $b \in \{0,1\}$。如果 $b = 0$，设置 $\hat{k} := k$。如果 $b = 1$，则选择一个均匀的 $\hat{k} \in \{0,1\}^n$。
3. 将 $(pk, c, \hat{k})$ 给予 $\mathcal{A}$，$\mathcal{A}$ 可以访问预言机 $Decaps_{sk}(\cdot)$，但**不得请求**解封装 c 本身。
4. $\mathcal{A}$ 输出一个比特 b'。如果 $b' = b$，则实验的输出定义为 1，否则为 0。

定义 11.13 如果对于所有概率多项式时间对手 $\mathcal{A}$，存在一个可忽略函数 negl，使得

$$Pr[\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

则密钥封装机制 Π 是 **CCA-安全**的。

幸运的是，我们可以证明使用 CCA-安全 KEM 与 CCA-安全私钥加密方案的组合会产生一个**抵抗选择密文攻击的公钥加密方案**。

定理 11.14 如果 Π 是一个 CCA-安全 KEM 并且 Π' 是一个 CCA-安全私钥加密方案，则如构造 11.10 中的 Π^{hy} 是一个 **CCA-安全公钥加密方案**。

证明是通过对定理 11.12 的证明进行适当修改获得的。

11.4 基于 CDH/DDH 的加密

到目前为止，我们已经抽象地讨论了公钥加密，但尚未看到任何公钥加密方案（或 KEM）的具体示例。在这里，我们探索一些基于 **Diffie-Hellman 问题**的构造。（Diffie-Hellman 问题在 8.3.2 节中介绍。）

11.4.1 El Gamal 加密

1985 年，Taher El Gamal 观察到 Diffie-Hellman 密钥交换协议（参见 10.3 节）可以用于提供一个公钥加密方案。回想一下，在 Diffie-Hellman 协议中，爱丽丝向鲍勃发送一条消息，然后鲍勃回复一条消息给爱丽丝；基于这些消息，爱丽丝和鲍勃可以导出一个共享值 k，该值与某个群 $\mathbb{G}$ 的均匀元素不可区分（对窃听者而言）。我们可以想象鲍勃使用该共享值来加密消息

$m \in \mathbb{G}$，方法是简单地将 $k \cdot m$ 发送给爱丽丝；爱丽丝显然可以使用她对 k 的知识来恢复 m，我们将在下面论证窃听者对 m 一无所知。

在 El Gamal 加密方案中，我们简单地**改变**了对上述交互的看法。我们将**爱丽丝的初始消息**视为她的**公钥**，将**鲍勃的回复**（她的初始回复和 $k \cdot m$）视为**密文**。基于**判定 Diffie-Hellman (DDH) 假设**的 CPA-安全性可以从 Diffie-Hellman 密钥交换协议的安全性（定理 10.3）相当容易地得出。

在我们的形式化处理中，我们首先陈述并证明一个简单的引理，该引理是 El Gamal 加密方案的基础。令 $\mathbb{G}$ 是一个有限群，令 $m \in \mathbb{G}$ 是一个任意元素。该引理指出，用**均匀群元素** k 乘以 m 产生的群元素 k' 是**均匀分布**的。重要的是，k' 的分布**独立于** m；这意味着 k' 不包含有关 m 的任何信息。

引理 11.15 令 $\mathbb{G}$ 是一个有限群，令 $m \in \mathbb{G}$ 是任意的。那么选择均匀 $k \in \mathbb{G}$ 并设置 $k' := k \cdot m$ 得到的 k' 的分布与选择均匀 $k' \in \mathbb{G}$ 得到的分布**相同**。换句话说，对于任何 $\hat{g} \in \mathbb{G}$，我们有

$$Pr[k \cdot m = \hat{g}] = 1/|\mathbb{G}|,$$

其中概率是取自对 $k \in \mathbb{G}$ 的均匀选择。

证明 令 $\hat{g} \in \mathbb{G}$ 是任意的。那么

$$Pr[k \cdot m = \hat{g}] = Pr[k = \hat{g} \cdot m^{-1}].$$

由于 k 是均匀的，因此 k 等于固定元素 $\hat{g} \cdot m^{-1}$ 的概率恰好是 $1/|\mathbb{G}|$。

上述引理提出了一种构造消息空间为 $\mathbb{G}$ 的**完美秘密私钥加密方案**的方法。发送方和接收方共享一个均匀元素 $k \in \mathbb{G}$ 作为他们的秘密密钥。为了加密消息 $m \in \mathbb{G}$，发送方计算密文 $k' := k \cdot m$。接收方可以通过计算 $m := k'/k$ 从密文 k' 恢复消息。完美秘密性立即从上述引理得出。事实上，我们已经以不同的形式看到了这个方案——**一次性密码本**加密方案是这种方法的实例化，其中底层群是某种固定长度的位串集，操作是按位异或。

我们可以通过为双方提供一种**通过公共信道交互**来生成一个共享的、“看起来随机”的值 k 的方法，从而将上述想法应用于公钥设置。这听起来应该很熟悉，因为它正是 Diffie-Hellman 协议所提供的。我们继续介绍细节。

与 8.3.2 节中一样，令 $\mathcal{G}$ 是一个**多项式时间**算法，它以 1^n 为输入，并（除了可能以可忽略的概率）输出一个**循环群** $\mathbb{G}$ 的描述、其**阶** q（其中 $||q|| = n$）和一个**生成元** g。El Gamal 加密

方案在构造 11.16 中描述。

构造 11.16

令 $\mathcal{G}$ 如正文所述。按如下方式定义公钥加密方案：

- **Gen**：在输入 1^n 时，运行 $\mathcal{G}(1^n)$ 以获得 $(\mathbb{G}, q, g)$。然后选择一个均匀的 $x \in \mathbb{Z}_q$ 并计算 $h := g^x$。公钥是 $\langle \mathbb{G}, q, g, h \rangle$，私钥是 $\langle \mathbb{G}, q, g, x \rangle$。消息空间是 $\mathbb{G}$。
- **Enc**：在输入公钥 $pk = \langle \mathbb{G}, q, g, h \rangle$ 和消息 $m \in \mathbb{G}$ 时，选择一个均匀的 $y \in \mathbb{Z}_q$ 并输出密文

$$\langle g^y, h^y \cdot m \rangle.$$

- **Dec**：在输入私钥 $sk = \langle \mathbb{G}, q, g, x \rangle$ 和密文 $\langle c_1, c_2 \rangle$ 时，输出

$$\hat{m} := c_2 / c_1^x.$$

El Gamal 加密方案。

为了证明解密成功，令 $\langle c_1, c_2 \rangle = \langle g^y, h^y \cdot m \rangle$，其中 $h = g^x$。那么

$$\hat{m} = \frac{c_2}{c_1^x} = \frac{h^y \cdot m}{(g^y)^x} = \frac{(g^x)^y \cdot m}{g^{xy}} = \frac{g^{xy} \cdot m}{g^{xy}} = m.$$

示例 11.17

令 $q = 83$ 且 $p = 2q + 1 = 167$，令 $\mathbb{G}$ 表示模 p 的二次剩余（即平方）群。（由于 p 和 q 是素数，$\mathbb{G}$ 是 $\mathbb{Z}_p^*$ 的一个阶为 q 的子群。参见 8.3.3 节。）由于 $\mathbb{G}$ 的阶是素数，因此 $\mathbb{G}$ 中除 1 以外的任何元素都是生成元；取 $g = 2^2 = 4 \bmod 167$。假设接收方选择秘密密钥 $37 \in \mathbb{Z}_{83}$，因此公钥为

$$pk = \langle p, q, g, h \rangle = \langle 167, 83, 4, [4^{37} \bmod 167] \rangle = \langle 167, 83, 4, 76 \rangle$$

其中我们使用 p 来表示 $\mathbb{G}$（假设接收方知道该群是模 p 的二次剩余集合）。

假设发送方加密消息 $m = 65 \in \mathbb{G}$（请注意 $65 = 30^2 \bmod 167$，因此 65 是子群中的一个元素）。如果 $y = 71$，则密文是

$$([4^{71} \bmod 167], [76^{71} \cdot 65 \bmod 167]) = \langle 132, 44 \rangle.$$

要解密，接收方首先计算 $124 = [132^{37} \bmod 167]$；然后，由于 $66 = [124^{-1} \bmod 167]$，接收方恢复 $m = 65 = [44 \cdot 66 \bmod 167]$。

我们现在证明该方案的安全性。（读者可能希望将以下证明与定理 3.18 和 10.3 的证明进行比较。）

定理 11.18 如果 DDH 问题相对于 $\mathcal{G}$ 是困难的，则 El Gamal 加密方案是 **CPA-安全**的。

证明 令 Π 表示 El Gamal 加密方案。我们证明 Π 在窃听者存在下具有不可区分加密；根据命题 11.3，这蕴含它是 CPA-安全的。

令 $\mathcal{A}$ 是一个概率多项式时间对手。我们想证明存在一个可忽略函数 negl，使得

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

考虑修改后的"加密方案" $\tilde{\Pi}$，其中 **Gen** 与 Π 中相同，但相对于公钥 $\langle \mathbb{G}, q, g, h\rangle$ 对消息 m 的加密是通过选择均匀 $y, z \in \mathbb{Z}_q$ 并输出密文

$$\langle g^y, g^z \cdot m\rangle.$$

虽然 $\tilde{\Pi}$ 实际上不是一个加密方案（因为接收方无法解密），但实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\tilde{\Pi}}(n)$ 仍然定义良好，因为该实验仅取决于密钥生成和加密算法。

引理 11.15 及其紧随其后的讨论表明，方案 $\tilde{\Pi}$ 中的密文的**第二个组件**是一个**均匀分布**的群元素，特别是**独立于**被加密的消息 m。（请记住，当 z 从 $\mathbb{Z}_q$ 中均匀选择时，g^z 是 $\mathbb{G}$ 的一个均匀元素。）密文的第一个组件显然独立于 m。总而言之，这意味着**整个密文不包含有关 m 的任何信息**。由此可知

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\tilde{\Pi}}(n) = 1] = \frac{1}{2}.$$

现在考虑以下试图解决相对于 $\mathcal{G}$ 的 DDH 问题的 PPT 算法 $\mathcal{D}$。回想一下，$\mathcal{D}$ 接收 $\langle \mathbb{G}, q, g, h_1, h_2, h_3\rangle$，其中 $h_1 = g^x, h_2 = g^y$，而 h_3 要么是 g^{xy} 要么是 g^z（对于均匀 x, y, z）；$\mathcal{D}$ 的目标是确定是哪种情况。

算法 $\mathcal{D}$：

该算法以 $\langle \mathbb{G}, q, g, h_1, h_2, h_3\rangle$ 为输入。

- 设置 $pk = \langle \mathbb{G}, q, g, h_1\rangle$ 并运行 $\mathcal{A}(pk)$ 以获得两条消息 $m_0, m_1 \in \mathbb{G}$。
- 选择一个均匀比特 b，设置 $c_1 := h_2$ 和 $c_2 := h_3 \cdot m_b$。
- 将密文 $\langle c_1, c_2\rangle$ 给予 $\mathcal{A}$ 并获得输出比特 b'。
- 如果 $b' = b$，输出 1；否则，输出 0。

我们来分析 $\mathcal{D}$ 的行为。有两种情况需要考虑：

情况 1： 假设 $\mathcal{D}$ 的输入是通过运行 $\mathcal{G}(1^n)$ 获得 $\langle \mathbb{G}, q, g \rangle$，然后选择均匀 $x, y, z \in \mathbb{Z}_q$，最后设置 $h_1 := g^x, h_2 := g^y, h_3 := g^z$ 来生成的。那么 $\mathcal{D}$ 在构造的公钥上运行 $\mathcal{A}$

$$pk = \langle \mathbb{G}, q, g, g^x \rangle$$

以及构造的密文

$$\langle c_1, c_2 \rangle = \langle g^y, g^z \cdot m_b \rangle.$$

我们看到，在这种情况下，在 $\mathcal{D}$ 中作为子程序运行的 $\mathcal{A}$ 的视图与 $\mathcal{A}$ 在实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\tilde{\Pi}}(n)$ 中的视图**分布相同**。由于 $\mathcal{D}$ 恰好在 $\mathcal{A}$ 的输出 b' 等于 b 时输出 1，我们有

$$Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] = Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\tilde{\Pi}}(n) = 1] = \frac{1}{2}.$$

情况 2： 假设 $\mathcal{D}$ 的输入是通过运行 $\mathcal{G}(1^n)$ 获得 $\langle \mathbb{G}, q, g \rangle$，然后选择均匀 $x, y \in \mathbb{Z}_q$，最后设置 $h_1 := g^x, h_2 := g^y, h_3 := g^{xy}$ 来生成的。那么 $\mathcal{D}$ 在构造的公钥上运行 $\mathcal{A}$

$$pk = \langle \mathbb{G}, q, g, g^x \rangle$$

以及构造的密文

$$\langle c_1, c_2 \rangle = \langle g^y, g^{xy} \cdot m_b \rangle = \langle g^y, (g^x)^y \cdot m_b \rangle.$$

我们看到，在这种情况下，在 $\mathcal{D}$ 中作为子程序运行的 $\mathcal{A}$ 的视图与 $\mathcal{A}$ 在实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)$ 中的视图**分布相同**。由于 $\mathcal{D}$ 恰好在 $\mathcal{A}$ 的输出 b' 等于 b 时输出 1，我们有

$$Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1] = Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1].$$

在 DDH 问题相对于 $\mathcal{G}$ 是困难的假设下，存在一个可忽略函数 negl 使得

$$\mathrm{negl}(n) \geq |Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1]| = |\frac{1}{2} - Pr[\mathrm{PubK}$$

这蕴含 $Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$，从而完成了证明。

El Gamal 实现问题

我们简要讨论一些与 El Gamal 加密相关的实际问题。

共享公共参数。 构造 11.16 中对 El Gamal 加密方案的描述要求接收方运行 $\mathcal{G}$ 来生成 $\mathbb{G}, q, g$

。在实践中，通常会生成并“一劳永逸”地固定这些参数，然后由多个接收方共享。（当然，每个接收方必须选择自己的秘密值 x 并发布自己的公钥 $h = g^x$。）例如，NIST 发布了一套推荐参数，适用于 El Gamal 加密方案。以这种方式共享参数**不会**影响安全性（假设参数最初是正确和诚实地生成的）。展望未来，我们指出这与 **RSA** 的情况**形成对比**，RSA 中的参数无法安全地共享（参见 11.5.1 节）。

群的选择。 正如 8.3.2 节中所讨论的，群阶 q 通常选择为**素数**。就具体群而言，**椭圆曲线**是一种越来越流行的选择；另一种选择是令 $\mathbb{G}$ 是 $\mathbb{Z}_p^*$ 的素数阶子群，其中 p 是素数。我们参考 9.3 节以获取实现不同安全级别的推荐密钥长度的列表。

消息空间。 El Gamal 加密方案的一个不方便之处在于，消息空间是群 $\mathbb{G}$ 而不是某个指定长度的位串。对于某些群的选择，可以通过定义位串到群元素的可逆编码来解决这个问题。在这种情况下，发送方可以首先将其消息 $m \in \{0,1\}^l$ 编码为群元素 $\hat{m} \in \mathbb{G}$，然后对 $\hat{m}$ 应用 El Gamal 加密。接收方可以像构造 11.16 中那样解密以获得编码消息 $\hat{m}$，然后反转编码以恢复原始消息 m。

一种更简单的方法是使用 El Gamal 加密（的一种变体）作为**混合加密方案的一部分**。例如，发送方可以选择一个均匀群元素 $m \in \mathbb{G}$，使用 El Gamal 加密方案对其进行加密，然后使用私钥加密方案和密钥 $H(m)$ 加密其实际消息，其中 $H : \mathbb{G} \to \{0,1\}^n$ 是一个合适的**密钥派生函数**（key-derivation function，参见下一节）。在这种情况下，使用我们接下来描述的**基于 DDH 的 KEM** 将更高效。

11.4.2 基于 DDH 的密钥封装

在上一节的末尾，我们指出 El Gamal 加密可以用作混合加密方案的一部分，方法是简单地加密一个均匀群元素 m 并使用该元素的哈希作为密钥。但这太浪费了！El Gamal 加密的安全性证明表明 c_1^x（其中 c_1 是密文的第一个组件， x 是接收方的私钥）**已经**与均匀群元素不可区分，因此发送方/接收方不妨使用它。构造 11.19 说明了遵循这种方法的 KEM。请注意，由此产生的封装仅包含**单个**群元素。相比之下，如果我们要对均匀群元素使用 El Gamal 加密，密文将包含**两个**群元素。

构造 11.19

令 $\mathcal{G}$ 如上一节所述。按如下方式定义 KEM：

- **Gen**：在输入 1^n 时，运行 $\mathcal{G}(1^n)$ 以获得 $(\mathbb{G}, q, g)$。选择一个均匀的 $x \in \mathbb{Z}_q$ 并设置 $h := g^x$。还指定一个函数 $H : \mathbb{G} \to \{0,1\}^{l(n)}$（对于某个函数 l，参见正文）。公钥是 $\langle \mathbb{G}, q, g, h, H \rangle$，私钥是 $\langle \mathbb{G}, q, g, x \rangle$。

- **Encaps**：在输入公钥 $pk = \langle \mathbb{G}, q, g, h, H \rangle$ 时，选择一个均匀的 $y \in \mathbb{Z}_q$ 并输出密文 g^y 和密钥 $H(h^y)$。
- **Decaps**：在输入私钥 $sk = \langle \mathbb{G}, q, g, x \rangle$ 和密文 $c \in \mathbb{G}$ 时，输出密钥 $H(c^x)$。

一个“类 El Gamal” KEM。

如上所述，该构造未指定密钥派生函数 H，并且有几种选择。（有关密钥派生的更多信息，请参见 5.6.4 节。）一种可能性是选择一个函数 $H : \mathbb{G} \to \{0,1\}^l$，它是（接近）**正则**的（regular），这意味着对于每个可能的密钥 $k \in \{0,1\}^l$，映射到 k 的群元素数量大致相同。（形式上，我们需要一个可忽略函数 negl，使得对于每个 $k \in \{0,1\}^l$

$$2^l \cdot |Pr[H(g) = k] - 2^{-l}| \leq \mathrm{negl}(n),$$

其中概率取自对 $g \in \mathbb{G}$ 的均匀选择。这确保了密钥 k 的分布在统计上接近均匀。）H 的复杂性以及可实现的密钥长度 l 将取决于所使用的具体群 $\mathbb{G}$。

第二种可能性是令 H 是一个**带密钥的函数**，其中 H 的（均匀）密钥作为接收方公钥的一部分包含在内。如果 H 是一个**强提取器**（strong extractor）（如 5.6.4 节中简要提及），则这可行。此处 l 的适当选择（以确保生成的密钥在统计上接近均匀）将取决于 $\mathbb{G}$ 的大小。

在上述任何一种情况下，基于**判定 Diffie-Hellman (DDH) 假设**的 CPA-安全性证明都可以通过调整 Diffie-Hellman 密钥交换协议（定理 10.3）的安全性证明而轻易得出。

定理 11.20 如果 DDH 问题相对于 $\mathcal{G}$ 是困难的，并且 H 是按所述方式选择的，则构造 11.19 是一个 **CPA-安全 KEM**。

如果愿意将 H 建模为**随机预言机**，则构造 11.19 可以基于（较弱的）**计算 Diffie-Hellman (CDH) 假设**被证明是 CPA-安全的。我们将在下一节讨论这个问题。

11.4.3 *随机预言机模型中的基于 CDH 的 KEM

在本节中，我们证明如果愿意将 H 建模为**随机预言机**，则构造 11.19 可以基于 **CDH 假设**被证明是 CPA-安全的。（读者可能希望回顾 5.5 节以提醒自己随机预言机模型。）直觉上，CDH 假设意味着观察到 $h = g^x$（来自公钥）和密文 $c = g^y$ 的攻击者无法计算 $DH_g(h, c) = h^y$。特别是，攻击者无法向随机预言机查询 h^y。但这又意味着从攻击者的角度来看，封装的密钥 $H(h^y)$ 是**完全随机**的。这种直觉在下面的形式证明中得到了体现。

如上述直觉所示，证明本质上依赖于将 H 建模为**随机预言机**。具体而言，证明依赖于以下事实：(1) 了解 $H(h^y)$ 的**唯一**方法是向 H 明确查询 h^y，这意味着攻击者已经解决了 CDH 实例

（这在 5.5.1 节中称为“可提取性”），以及 (2) 如果攻击者不查询 h^y 到 H，那么从攻击者的角度来看，$H(h^y)$ 的值是**均匀**的。这些属性只有在 H 被建模为随机预言机时才成立——事实上，它们也只有在建模为随机预言机时才有意义。

定理 11.21 如果 CDH 问题相对于 $\mathcal{G}$ 是困难的，并且 H 被建模为**随机预言机**，则构造 11.19 是 **CPA-安全**的。

证明 令 Π 表示构造 11.19，令 $\mathcal{A}$ 是一个 PPT 对手。我们想证明存在一个可忽略函数 negl，使得

$$Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

上述概率也取自对函数 H 的均匀选择，$\mathcal{A}$ 被授予对其的预言机访问权限。

考虑实验 $\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$ 的一次执行，其中公钥是 $\langle \mathbb{G}, q, g, h \rangle$，密文是 $c = g^y$，令 Query 是 $\mathcal{A}$ 查询 $DH_g(h, c) = h^y$ 到 H 的事件。我们有

$$Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1] = Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathrm{Query}}] + Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 \wedge \mathrm{Query}] \leq P$$

如果 $Pr[\mathrm{Query}] = 0$，则 $Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathrm{Query}}] = 0$。否则，

$$Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathrm{Query}}] = Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 | \overline{\mathrm{Query}}] \cdot Pr[\overline{\mathrm{Query}}] \leq Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n$$

在实验 $\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$ 中，对手 $\mathcal{A}$ 被给予公钥和密文，加上**要么**与该密文对应的封装密钥 $k = H(h^y)$，**要么**一个均匀密钥。如果 Query 没有发生，那么从对手的角度来看，k 是**均匀分布**的，因此 $\mathcal{A}$ 无法区分这两种可能性。这意味着

$$Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1 | \overline{\mathrm{Query}}] = \frac{1}{2}.$$

回到等式 (11.18)，我们因此有

$$Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + Pr[\mathrm{Query}].$$

我们接下来证明 $Pr[\mathrm{Query}]$ 是可忽略的，从而完成了证明。

令 $t = t(n)$ 是 $\mathcal{A}$ 对随机预言机 H 进行的查询次数的（多项式）上限。定义以下用于 CDH 问题的 PPT 算法 $\mathcal{A}'$（相对于 $\mathcal{G}$）：

算法 $\mathcal{A}'$：

该算法以 $\mathbb{G}, q, g, h, c$ 为输入。

- 设置 $pk = \langle \mathbb{G}, q, g, h \rangle$ 并选择一个均匀的 $k \in \{0,1\}^l$。
- 运行 $\mathcal{A}(pk, c, k)$。当 $\mathcal{A}$ 对 H 进行查询时，通过选择一个新的均匀 l-比特串来回答。
- 在 $\mathcal{A}$ 的执行结束时，令 $y_1, \ldots, y_t$ 是 $\mathcal{A}$ 对 H 进行的查询列表。选择一个均匀索引 $i \in \{1, \ldots, t\}$ 并输出 y_i。

我们对 $\mathcal{A}'$ 解决 CDH 问题的概率感兴趣，即 $Pr[\mathcal{A}'(\mathbb{G}, q, g, h, c) = DH_g(h, c)]$。为了分析这个概率，首先请注意，事件 Query 在 $\mathcal{A}'$ 的执行中仍然定义良好，尽管 $\mathcal{A}'$ 无法检测到它是否发生。此外，当 $\mathcal{A}$ 在 $\mathcal{A}'$ 中作为子程序运行时，事件 Query 的概率与实验 $\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$ 中事件 Query 的概率**相同**。这是因为 $\mathcal{A}$ 的视图在两种情况下是相同的，直到事件 Query 发生：在每种情况下，$\mathbb{G}, q, g$ 都是由 $\mathcal{G}(1^n)$ 输出的；在每种情况下，h 和 c 都是 $\mathbb{G}$ 的均匀元素，k 是一个均匀的 l-比特串；在每种情况下，对 H 的查询（除了 $H(DH_g(h, c))$）都以一个均匀的 l-比特串回答。（在 $\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$ 中，查询 $H(DH_g(h, c))$ 以实际的封装密钥回答，该密钥等于 k 的概率为 $1/2$；而在 $\mathcal{A}'$ 中作为子程序运行时，查询 $H(DH_g(h, c))$ 以一个独立于 k 的均匀 l-比特串回答。但是当进行此查询时，事件 Query 发生。）

最后，观察到当 Query 发生时，根据定义 $DH_g(h, c) \in \{y_1, \ldots, y_t\}$，因此 $\mathcal{A}'$ 以至少 $1/t$ 的概率输出正确结果 $DH_g(h, c)$。因此，我们得出结论

$$Pr[\mathcal{A}'(\mathbb{G}, q, g, h, c) = DH_g(h, c)] \geq Pr[\mathrm{Query}]/t$$

或 $Pr[\mathrm{Query}] \leq t \cdot Pr[\mathcal{A}'(\mathbb{G}, q, g, h, c) = DH_g(h, c)]$。由于 CDH 问题对于 $\mathcal{G}$ 是困难的，后者的概率是**可忽略的**；由于 t 是多项式，这蕴含 $Pr[\mathrm{Query}]$ 也是**可忽略的**。这完成了证明。

在下一节中，我们将看到构造 11.19 可以被证明在更强的 CDH 假设变体下是 CCA-安全的（如果我们继续将 H 建模为随机预言机）。

11.4.4 选择密文安全性与 DHIES/ECIES

El Gamal 加密方案容易受到**选择密文攻击**。这是因为它具有**可塑性**。回想一下，非正式地，如果给定某个未知消息 m 的加密 c，可以生成一个修改后的密文 c'，它是消息 m' 的加密，其中 m' 与 m 存在某种**已知**关系，则加密方案是可塑的。

在 El Gamal 加密的情况下，考虑一个对手 $\mathcal{A}$ 拦截密文 $c = \langle c_1, c_2 \rangle$，该密文是使用公钥 $pk = \langle \mathbb{G}, q, g, h \rangle$ 加密的，然后 $\mathcal{A}$ 构造修改后的密文 $c' = \langle c_1, c_2' \rangle$，其中 $c_2' = c_2 \cdot \alpha$（对于某个 $\alpha \in \mathbb{G}$）。如果 c 是消息 $m \in \mathbb{G}$ 的加密（$\mathcal{A}$ 可能未知），我们有 $c_1 = g^y$ 和 $c_2 =$

$h^y \cdot m$（对于某个 $y \in \mathbb{Z}_q$）。但是，

$$c_1 = g^y \quad \text{且} \quad c_2' = h^y \cdot (\alpha \cdot m),$$

因此 c' 是消息 $\alpha \cdot m$ 的**有效加密**。换句话说，$\mathcal{A}$ 可以将（未知）消息 m 的加密转换为（未知）消息 $\alpha \cdot m$ 的加密。正如 11.2.3 节中场景 3 所讨论的，这种攻击可能会产生严重的后果。

上一节中讨论的 KEM 也可能是可塑的，这取决于所使用的具体密钥派生函数 H。然而，如果将 H 建模为**随机预言机**，则此类攻击似乎不再可能。事实上，在这种情况下，可以证明构造 11.19 基于所谓的 **Gap-CDH 假设**是 CCA-安全的。回想一下，CDH 假设说给定群元素 g^x 和 g^y（对于某个生成元 g），计算 g^{xy} 是不可行的。**Gap-CDH 假设**说，即使在给定一个预言机 $\mathcal{O}$（使得 $\mathcal{O}(U, V)$ 恰好在 $V = U^y$ 时返回 1）的情况下，这仍然是不可行的。换句话说，即使在给定一个解决 DDH 问题的预言机的情况下，CDH 问题仍然是困难的。（我们不给出正式定义，因为我们不会在本书的其余部分使用此假设。）我们相信此假设对于我们本书中讨论的群类别成立。以下定理的证明与定理 11.38 的证明非常相似。

定理 11.22 如果 Gap-CDH 问题相对于 $\mathcal{G}$ 是困难的，并且 H 被建模为**随机预言机**，则构造 11.19 是 **CCA-安全 KEM**。

有趣的是，可以分析同一构造（即构造 11.19）在不同假设和不同模型下的情况，从而产生不同的结果。仅假设 DDH 问题是困难的（并且对于 H 选择适当），该方案是 CPA-安全的。如果我们模型 H 为随机预言机（这在 H 上施加了更严格的要求），那么在较弱的 CDH 假设下，我们获得 CPA-安全性，而在较强的 Gap-CDH 假设下，我们获得 CCA-安全性。

构造 11.23

令 $\mathcal{G}$ 如正文所述。令 $\Pi_E = (Enc', Dec')$ 是一个私钥加密方案，令 $\Pi_M = (Mac, Vrfy)$ 是一个消息认证码。按如下方式定义公钥加密方案：

- **Gen**：在输入 1^n 时，运行 $\mathcal{G}(1^n)$ 以获得 $(\mathbb{G}, q, g)$。选择一个均匀的 $x \in \mathbb{Z}_q$，设置 $h := g^x$ 并指定一个函数 $H : \mathbb{G} \to \{0,1\}^{2n}$。公钥是 $\langle \mathbb{G}, q, g, h, H \rangle$，私钥是 $\langle \mathbb{G}, q, g, x, H \rangle$。
- **Enc**：在输入公钥 $pk = \langle \mathbb{G}, q, g, h, H \rangle$ 时，选择一个均匀的 $y \in \mathbb{Z}_q$ 并设置 $k_E || k_M := H(h^y)$。计算 $c' \leftarrow Enc'_{k_E}(m)$，并输出密文 $\langle g^y, c', Mac_{k_M}(c') \rangle$。
- **Dec**：在输入私钥 $sk = \langle \mathbb{G}, q, g, x, H \rangle$ 和密文 $\langle c, c', t \rangle$ 时，如果 $c \notin \mathbb{G}$ 则输出 $\perp$。否则，计算 $k_E || k_M := H(c^x)$。如果 $Vrfy_{k_M}(c', t) \neq 1$ 则输出 $\perp$；否则，输出 $Dec'_{k_E}(c')$。

DHIES/ECIES。

使用构造 11.19 的 CCA-安全加密。 将构造 11.19 中的 KEM 与任何 CCA-安全私钥加密方案相结合，即可产生一个 CCA-安全公钥加密方案。（参见定理 11.14。）使用构造 4.18 作为私钥组件实例化此方法，与 DHIES/ECIES 中所做的匹配，DHIES/ECIES 的变体包含在 ISO/IEC 18033-2 公钥加密标准中。（参见构造 11.23。）这些方案中对消息 m 的加密采用以下形式

$$\langle g^y, Enc'_{k_E}(m), Mac_{k_M}(c')\rangle,$$

其中 Enc' 表示一个 CPA-安全私钥加密方案，c' 表示 $Enc'_{k_E}(m)$。DHIES（Diffie-Hellman 集成加密方案）可以泛指任何这种形式的方案，或特指群 $\mathbb{G}$ 是有限域的循环子群的情况。ECIES（椭圆曲线集成加密方案）指 $\mathbb{G}$ 是椭圆曲线群的情况。我们指出，在构造 11.23 中，在解密过程中检查 c（密文的第一个组件）是否在 $\mathbb{G}$ 中至关重要。否则，攻击者可能会请求解密一个畸形的密文 $\langle c, c', t\rangle$，其中 $c \notin \mathbb{G}$；解密此类密文（即不返回 $\perp$）可能会泄露有关私钥的信息。

根据定理 4.19，加密消息然后应用（强）消息认证码会产生一个 CCA-安全私钥加密方案。结合定理 11.14，我们得出结论：

推论 11.24 令 Π_E 是一个 CPA-安全私钥加密方案，令 Π_M 是一个强安全的**消息认证码**。如果 Gap-CDH 问题相对于 $\mathcal{G}$ 是困难的，并且 H 被建模为**随机预言机**，则构造 11.23 是一个**CCA-安全公钥加密方案**。

11.5 RSA 加密

在本节中，我们将注意力转向基于 8.2.4 节中定义的 **RSA 假设**的加密方案。我们指出，尽管基于 RSA 的加密目前正在广泛使用，但目前也正在**逐渐**从使用 RSA 转向使用基于 CDH/DDH 的密码系统（依赖于椭圆曲线群），因为基于 RSA 的方案需要更长的密钥长度。我们参考 9.3 节以获取进一步的讨论。

11.5.1 纯 RSA

我们首先描述一个基于 **RSA 问题**的简单加密方案。尽管该方案**不安全**，但它为后续的安全方案提供了一个有用的起点。

令 GenRSA 是一个 **PPT 算法**，它在输入 1^n 时，输出一个模数 N，它是两个 n-比特素数的乘积，以及满足 $ed \equiv 1 \bmod \phi(N)$ 的整数 e, d。（像往常一样，该算法可能会以可忽略的概

率失败，但我们在此忽略。）回想一下 8.2.4 节，这样的算法可以很容易地从任何输出复合模数 N 及其因式分解的算法 $\mathrm{GenModulus}$ 构造出来；参见算法 11.25。

算法 11.25

RSA 密钥生成 GenRSA

输入： 安全参数 1^n **输出：** N, e, d 如正文所述

1. $(N, p, q) \leftarrow \mathrm{GenModulus}(1^n)$
2. $\phi(N) := (p-1)(q-1)$
3. 选择 $e > 1$ 使得 $gcd(e, \phi(N)) = 1$
4. 计算 $d := [e^{-1} \bmod \phi(N)]$
5. 返回 N, e, d

令 N, e, d 如上，令 $c = m^e \bmod N$。RSA 加密依赖于以下事实：知道 d 的人可以通过计算 $[c^d \bmod N]$ 从 c 恢复 m；这之所以有效，是因为

$$c^d = (m^e)^d = m^{ed} \equiv m \bmod N,$$

正如 8.2.4 节中所讨论的。另一方面，在不知道 d 的情况下（即使知道 N 和 e），RSA 假设（参见定义 8.46）意味着从 c 恢复 m 是困难的，至少如果 m 是从 $\mathbb{Z}_N^*$ 中均匀选择的。这自然提出了构造 11.26 中所示的公钥加密方案：接收方运行 GenRSA 以获得 N, e, d；它发布 N 和 e 作为其公钥，并将 d 保留在其私钥中。为了加密消息 $m \in \mathbb{Z}_N^*$，发送方计算密文 $c := [m^e \bmod N]$。正如我们刚刚指出的，知道 d 的接收方可以解密 c 并恢复 m。

构造 11.26

令 GenRSA 如正文所述。按如下方式定义公钥加密方案：

- **Gen**：在输入 1^n 时，运行 $\mathrm{GenRSA}(1^n)$ 以获得 N, e 和 d。公钥是 $\langle N, e \rangle$，私钥是 $\langle N, d \rangle$。
- **Enc**：在输入公钥 $pk = \langle N, e \rangle$ 和消息 $m \in \mathbb{Z}_N^*$ 时，计算密文

$$c := [m^e \bmod N].$$

- **Dec**：在输入私钥 $sk = \langle N, d \rangle$ 和密文 $c \in \mathbb{Z}_N^*$ 时，计算消息

$$m := [c^d \bmod N].$$

纯 RSA 加密方案。

以下是上述方案的一个工作示例（另请参见示例 8.49）。

示例 11.27

假设 GenRSA 输出 $(N, e, d) = (391, 3, 235)$。（请注意 $391 = 17 \cdot 23$，因此 $\phi(391) = 16 \cdot 22 = 352$。此外，$3 \cdot 235 = 705 \equiv 1 \bmod 352$。）因此公钥是 $\langle 391, 3 \rangle$，私钥是 $\langle 391, 235 \rangle$。

为了使用公钥 $\langle 391, 3 \rangle$ 加密消息 $m = 158 \in \mathbb{Z}_{391}^*$，我们简单地计算 $c := [158^3 \bmod 391] = 295$；这就是密文。要解密，接收方计算 $[295^{235} \bmod 391] = 158$。

纯 RSA 加密方案安全吗？因式分解假设意味着，对于给定公钥的攻击者来说，推导出相应的私钥是**计算上不可行**的；参见 8.2.5 节。这对于公钥加密方案是**必要但不够**的。RSA 假设意味着，如果消息 m 是从 $\mathbb{Z}_N^*$ 中**均匀选择**的，那么给定 N, e 和 c（即公钥和密文）的窃听者**无法恢复** m。但这些是**弱保证**，远低于我们想要的安全级别！特别是，它们没有排除以下可能性：当消息不是从 $\mathbb{Z}_N^*$ 中均匀选择时（事实上，当 m 是从一个**小范围**中选择时，很容易看出攻击者可以从公钥和密文计算 m），攻击者可以恢复消息。此外，它没有排除攻击者可以学习关于消息的**部分信息**的可能性，即使它是均匀的（事实上，已知这是可能的）。此外，纯 RSA 加密是**确定性**的，因此必须是**不安全**的，正如我们已经在 11.2.1 节中讨论的那样。

纯 RSA 上的更多攻击

我们已经指出纯 RSA 加密不是 CPA-安全的。然而，可能有人会试图使用纯 RSA 来加密“随机消息”和/或在泄露消息的一些位信息是可以接受的情况下。我们通常不建议这样做，并在此仅提供一些可能出错的示例。（下面的一些攻击假设 $e = 3$。在某些情况下，攻击可以至少部分扩展到更大的 e；无论如何，正如 8.2.4 节中指出的，设置 $e = 3$ 在实践中经常完成。这些攻击应该被视为证明构造 11.26 是不充分的，而不是表明设置 $e = 3$ 必然是一个糟糕的选择。）

恢复 m 的二次改进。 由于纯 RSA 加密是确定性的，我们知道如果 $m < B$，那么攻击者可以在**时间** $\mathcal{O}(B)$ 内从密文 $c = [m^e \bmod N]$ 确定 m，方法是使用 11.2.1 节中讨论的暴力攻击。然而，人们可能希望如果 B 很大（即消息是从一个相当大的值集中选择的），则可以使用纯 RSA 加密。这可能发生的一个场景是在混合加密的背景下（参见 11.3 节），其中“消息”是一个随机的 n-比特密钥，因此 $B = 2^n$。不幸的是，存在一种巧妙的攻击，可以以**大约时间** $\mathcal{O}(\sqrt{B})$ 恢复 m，且概率很高。这在实践中可能会产生显着差异：一个 2^{80}-时间的攻击（例如）是不可行的，但一个在 2^{40} 时间内运行的攻击相对容易执行。

对该攻击的描述作为算法 11.28 给出。在我们的描述中，我们假设 $B = 2^n$ 并令 $\alpha \in (\frac{1}{2}, 1)$ 表示某个固定常数（见下文）。算法的时间复杂度主要取决于对 $2^{\alpha n}$ 对 (r, x_r) 进行排序所需

的时间；这可以在时间 $\mathcal{O}(n \cdot 2^{\alpha n})$ 内完成。在倒数第二行中使用二分查找来检查是否存在一个 r 使得 $x_r = [s^e \bmod N]$。

我们现在概述为什么该攻击以高概率恢复 m。令 $c = m^e \bmod N$。对于 $\alpha > \frac{1}{2}$ 的适当选择，可以证明，如果 m 是一个均匀的 n-比特整数，那么以高概率存在 r, s 使得 $1 < r \leq s \leq 2^{\alpha n}$ 且 $m = r \cdot s$。（例如，如果 $n = 64$，因此 m 是一个随机的 64 位串，那么以 0.35 的概率存在长度最多为 34 位的 r, s 使得 $m = r \cdot s$。有关详细信息，请参阅本章末尾的参考文献。）假设是这种情况，上述算法找到了 m，因为

$$c = m^e = (r \cdot s)^e = r^e \cdot s^e \bmod N,$$

因此 $x_r = c/r^e = s^e \bmod N$，且 $r, s < T$。

使用小 e 加密短消息。 前面的攻击表明如何在**时间约为** $\mathcal{O}(\sqrt{B})$ 内恢复已知小于某个界 B 的消息 m。在这里，我们证明如何在**时间** $poly(||N||)$ 内完成相同的操作，如果 $B \leq N^{1/e}$（其中这表示 N 的 e 次实数根）。该攻击依赖于以下观察：当 $m < N^{1/e}$ 时，将 m 提高到 e 次幂模 N **不涉及模约简**；即 $[m^e \bmod N]$ 等于整数 m^e。这意味着给定密文 $c = [m^e \bmod N]$，攻击者可以通过计算**整数**上的 $m := c^{1/e}$（即**不**取模 N）来确定 m；这可以在时间 $poly(||c||) = poly(||N||)$ 内轻松完成，因为求 e 次根在整数上很容易，只有在模 N 工作时才困难。

对于小 e，这代表了纯 RSA 加密的**严重缺陷**。例如，如果取 $e = 3$ 并假设 $||N|| \approx 1024$ 位，那么即使 m 是一个均匀的 300 位整数，该攻击也有效；这再次排除了纯 RSA 的安全性，即使将其用作混合加密方案的一部分。

加密部分已知消息。 此攻击可以视为前一个攻击的推广。它假设发送方加密一条消息，其中**一部分是已知**的（在使用安全加密方案时不应导致攻击）。在这里，我们依赖于 Coppersmith 的一个强大结果，我们不加证明地陈述：

定理 11.29 令 $p(x)$ 是一个 e 次多项式。那么可以在**时间** $poly(||N||, e)$ 内找到所有 m 使得 $p(m) \equiv 0 \bmod N$ 且 $|m| \leq N^{1/e}$。

由于运行时间对 e 的依赖性，该攻击仅对**小 e** 是实用的。在下文中，我们假设 $e = 3$ 以具体说明。

假设发送方将消息 $m = m_1||m_2$ 加密给接收方，公钥为 $\langle N, 3 \rangle$，其中消息的**第一部分 m_1 是已知**的，但第二部分 m_2 未知。具体来说，假设 m_2 长 k 位，因此 $m = B \cdot m_1 + m_2$，其中 $B = 2^k$。给定由此产生的密文 $c = [(m_1||m_2)^3 \bmod N]$，窃听者可以定义 $p(x) \triangleq (2^k \cdot m_1 + x)^3 - c$ 作为一个三次多项式。该多项式具有根 m_2（模 N），且 $|m_2| < B$。因

此，定理 11.29 蕴含，只要 $B \leq N^{1/3}$，攻击者就可以高效地计算 m_2。当 m_2 已知但 m_1 未知时，也有类似的攻击。

加密相关消息。 此攻击假设发送方将两条**相关**消息加密给同一接收方（在使用安全加密方案时不应导致攻击）。假设发送方将 m 和 $m+\delta$ 都加密给接收方，公钥为 $\langle N, e\rangle$，其中**偏移量 δ 是已知**的，但 m 未知。给定两条密文 $c_1 = [m^e \bmod N]$ 和 $c_2 = [(m+\delta)^e \bmod N]$，窃听者可以定义两个多项式 $f_1(x) \triangleq x^e - c_1$ 和 $f_2(x) \triangleq (x+\delta)^e - c_2$（模 N），每个多项式的次数都是 e。请注意 $x = m$ 是这两个多项式的一个根（模 N），因此线性项 $(x-m)$ 是它们两者共同的因子。因此，如果 $f_1(x)$ 和 $f_2(x)$ 的**最大公约数**（作为 $\mathbb{Z}_N^*$ 上的多项式）是线性的，它将揭示 m。最大公约数可以使用类似于附录 B.1.2 中所示的算法在时间 $poly(||N||, e)$ 内计算；因此，此攻击对于**小 e** 是可行的。

向多个接收方发送相同的消息。 我们的最后一次攻击假设发送方向**多个接收方**加密**相同的消息**（这再次不应在使用安全加密方案时导致攻击）。令 $e = 3$，并假设相同的消息 m 被加密给持有公钥 $pk_1 = \langle N_1, 3\rangle, pk_2 = \langle N_2, 3\rangle$ 和 $pk_3 = \langle N_3, 3\rangle$ 的三个不同方。假设 $gcd(N_i, N_j) = 1$（对于不同的 i, j）；如果不是，那么至少可以立即分解其中一个模数，并且可以轻松恢复消息 m。窃听者看到

$$c_1 = [m^3 \bmod N_1], \quad c_2 = [m^3 \bmod N_2], \quad \text{和} \quad c_3 = [m^3 \bmod N_3].$$

令 $N^* = N_1 N_2 N_3$。《中国剩余定理》的扩展版本说存在一个唯一的非负整数 $\hat{c} < N^*$ 使得

$$\hat{c} \equiv c_1 \bmod N_1, \quad \hat{c} \equiv c_2 \bmod N_2, \quad \text{和} \quad \hat{c} \equiv c_3 \bmod N_3.$$

此外，使用类似于 8.1.5 节中所示的技术，可以在给定公钥和上述密文的情况下高效地计算 $\hat{c}$。最后请注意，m^3 满足上述等式，且 $m^3 < N^*$，因为 $m < \min\{N_1, N_2, N_3\}$。与前一个攻击一样，这意味着 $\hat{c} = m^3$（在整数上，即**没有**发生模约简），因此可以通过计算 $\hat{c}$ 的整数立方根来恢复消息 m。

11.5.2 填充 RSA 和 PKCS #1 v1.5

尽管纯 RSA 是不安全的，但它确实提出了一种基于 RSA 问题的公钥加密的通用方法：要使用公钥 $\langle N, e\rangle$ 加密消息 m，首先将 m 映射到 $\mathbb{Z}_N^*$ 的一个元素 $\hat{m}$；然后计算密文 $c = [\hat{m}^e \bmod N]$。要解密密文 c，接收方计算 $\hat{m} = [c^d \bmod N]$，然后恢复原始消息 m。为了让接收方能够恢复消息，从消息到 $\mathbb{Z}_N^*$ 元素的映射必须是**（高效）可逆的**。为了让遵循这种方法的方案有希望是 CPA-安全的，该映射必须是**随机化**的，这样加密就不是确定性的。这当然是一个必要条件，但不是充分条件，加密方案的安全性**关键取决于所使用的具体映射**。

实现上述想法的一个简单方法是在加密之前**随机填充**消息。也就是说，为了将消息 m（被视为位串）映射到 $\mathbb{Z}_N^*$ 的一个元素，发送方选择一个均匀位串 $r \in \{0,1\}^l$（对于某个适当的 l）并设置 $\hat{m} := r||m$，由此产生的值可以自然地解释为 $\mathbb{Z}_N^*$ 中的一个整数，并且该映射显然是可逆的。参见构造 11.30。（对 $l(n)$ 和 m 长度的限制确保整数 $\hat{m}$ 小于 N。）

该构造由一个值 l 参数化，该值确定所使用的随机填充的长度。方案的安全性取决于 l。存在一个明显的暴力攻击，其运行时间为 2^l，因此如果 l **太短**（特别是如果 $l(n) = \mathcal{O}(\log n)$），则该方案是**不安全**的。在另一个极端，我们在下一节中证明（实质上）当填充**尽可能大**，且 m 只是一个比特时，可以基于 RSA 假设证明安全性。在中间情况下，情况不太清楚：对于某些 l 范围，我们无法基于 RSA 假设证明安全性，但也不知道多项式时间攻击。我们推迟进一步的讨论，直到我们接下来处理 PKCS #1 v1.5。

RSA PKCS #1 v1.5。 RSA 实验室公钥密码学标准 (PKCS) #1 版本 1.5（于 1993 年发布）利用了填充 RSA 加密的变体。对于通常形式的公钥 $pk = \langle N, e \rangle$，令 k 表示 N 的**字节长度**；即 k 是满足 $2^{8(k-1)} \leq N < 2^{8k}$ 的整数。要加密的消息 m 假定长度是 8 位的倍数，并且长度可以在 1 到 $k - 11$ 字节之间。对一个 D-字节长消息 m 的加密计算为

$$[(0\text{x}00||0\text{x}02||r||0\text{x}00||m)^e \bmod N]$$

其中 r 是一个随机生成的 $(k - D - 3)$-字节串，其字节**都不**等于 $0\text{x}00$。（后一个条件使得消息可以在解密时被**明确**恢复。）请注意，允许的 m 最大长度确保 r 的长度至少为 8 个字节。

不幸的是，按规范的 PKCS #1 v1.5 **不是 CPA-安全**的，因为它允许使用**太短的随机填充**。这最好通过展示攻击者可以确定已知具有许多尾随零的消息的初始部分来说明。为简单起见，假设 $m = b||\underbrace{0 \cdots 0}_{L}$，其中 $b \in \{0,1\}$ 是未知且 m 尽可能长（因此 $L = 8 \cdot (k - 11) - 1$）。对 m 的加密给出密文 c，其中

$$c = (0\text{x}00||0\text{x}02||r||0\text{x}00||b||0 \cdots 0)^e \bmod N.$$

攻击者可以计算 $c' = c/(2^L)^e \bmod N$；请注意

$$c' = \left(\frac{0\text{x}00||0\text{x}02||r||0\text{x}00||b||0 \cdots 0}{2^L}\right)^e = (0\text{x}00||0\text{x}02||r||0\text{x}00||b)^e \bmod N.$$

整数 $0\text{x}02||r||0\text{x}00||b$ 长 75 位（请注意 $0\text{x}02 = 0000\ 0010$，并且所有高阶 0 位都不计入），因此攻击者现在可以应用上一节中的“短消息攻击”或基于加密部分已知消息的攻击。为了避免这些攻击，我们需要取 r 的长度至少为 $||N||/e$。然而，即使 e 很大，上一节中的“二次改进攻击”表明 r 可以以大约 $2^{||r||/2}$ 的时间高概率恢复。

如果我们将 r 强制为大约 N 长度的一半，并相应地减少最大消息长度，那么合理地猜想 PKCS #1 v1.5 中的加密方案是 CPA-安全的。（然而，我们强调**没有**已知基于 RSA 假设的安全性证明。）然而，由于对该方案的**严重选择密文攻击**（在 11.5.5 节中简要描述），已引入了更新版本的 PKCS #1 标准，应该使用这些版本。

11.5.3 *没有随机预言机的 CPA-安全加密

在本节中，我们展示了一个可以被证明基于 **RSA 假设**是 **CPA-安全**的加密方案。我们首先描述 RSA 问题的特定**硬核谓词**（hard-core predicate）（参见 7.1.3 节），然后展示如何使用该硬核谓词来加密**单个比特**。然后我们扩展该方案以提供一个 KEM。

本节中描述的方案主要具有**理论兴趣**，**在实践中不使用**。这是因为它们的效率低于可以在**随机预言机模型**中被证明安全的其他基于 RSA 的构造（参见 5.5 节）。我们将在后续章节中看到此类加密方案的示例。

RSA 问题的硬核谓词。 松散地讲，RSA 假设说给定 N, e 和 $[x^e \bmod N]$（对于从 $\mathbb{Z}_N^*$ 中均匀选择的 x），恢复 x 是不可行的。就其本身而言，这并没有说明计算关于 x 的某些特定信息的计算难度。我们能否分离出关于 x 的一些**特定比特信息**，这些信息从 N, e 和 $[x^e \bmod N]$ 很难计算出来？**硬核谓词**的概念恰好捕捉了这一要求。（硬核谓词在 7.1.3 节中介绍。RSA 假设给出了一个单向排列族的事实将在 8.4.1 节中讨论。然而，我们在这里的处理是自包含的。）结果表明，x **的最低有效位**，表示为 $lsb(x)$，是 RSA 问题的一个硬核谓词。

为给定算法 GenRSA（具有通常行为）和算法 $\mathcal{A}$ 定义以下实验：

RSA 硬核谓词实验 $\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(1^n)$：

1. 运行 $\mathrm{GenRSA}(1^n)$ 以获得 (N, e, d)。
2. 选择一个均匀的 $x \in \mathbb{Z}_N^*$ 并计算 $y := [x^e \bmod N]$。
3. $\mathcal{A}$ 被给予 N, e, y 并输出一个比特 b。
4. 当且仅当 $lsb(x) = b$ 时，实验的输出为 1。

观察到，当 $x \in \mathbb{Z}_N^*$ 是均匀的时，$lsb(x)$ 是一个**均匀比特**。$\mathcal{A}$ 可以通过简单地输出一个均匀比特 b 以概率 $1/2$ 猜出 $lsb(x)$。以下定理指出，如果 RSA 问题是困难的，那么**任何高效**算法 $\mathcal{A}$ 都不能做得比这好得多；即，最低有效位是 RSA 排列的一个硬核谓词。

定理 11.31 如果 RSA 问题相对于 GenRSA 是困难的，则对于所有概率多项式时间算法 $\mathcal{A}$，存在一个可忽略函数 negl 使得 $Pr[\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$。

这个定理的完整证明超出了本书的范围。然而，我们通过**概述**一个较弱结果的证明来提供一些直

觉：RSA 假设蕴含对于所有概率多项式时间 $\mathcal{A}$，$Pr[\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n)=1]<1$。为了证明这一点，我们证明一个**高效**算法 $\mathcal{A}'$（它总是正确计算 $lsb(x)$）可以用于从 N,e 和 $[x^e \bmod N]$ 中**高效恢复** x（全部）。

固定 N 和 e，令 $\mathcal{A}$ 是一个算法，使得 $\mathcal{A}([r^e \bmod N])=lsb(r)$。给定 N,e 和 $y=[x^e \bmod N]$，我们将**逐位**恢复 x 的比特，从最低有效位到最高有效位。为了确定 $lsb(x)$，我们只需运行 $\mathcal{A}(y)$。现在有两种情况：

情况 1： $lsb(x)=0$。请注意 $y/2^e \equiv (x/2)^e \bmod N$，并且因为 x 是偶数（即 $lsb(x)=0$），所以 2 整除整数 x。因此 $x/2$ 只是 x 的右移一位，且 $lsb(x/2)$ 等于 x 的第二低有效位 $2sb(x)$。因此，我们可以通过计算 $y':=[y/2^e \bmod N]$，然后运行 $\mathcal{A}(y')$ 来获得 $2sb(x)$。

情况 2： $lsb(x)=1$。这里 $[x/2 \bmod N]=(x+N)/2$。因此 $lsb([x/2 \bmod N])$ 等于 $2sb(x+N)$；后者等于 $1\oplus 2sb(N)\oplus 2sb(x)$（我们在第二位有一个进位，因为 x 和 N 都是奇数）。因此，如果我们计算 $y':=[y/2^e \bmod N]$，那么 $2sb(x)=\mathcal{A}(y')\oplus 1\oplus 2sb(N)$。

以这种方式继续，我们可以恢复 x 的**所有**比特。

加密一个比特。 我们可以使用上面确定的硬核谓词来加密**单个比特**。想法很简单：要加密消息 $m\in\{0,1\}$，发送方选择均匀 $r\in\mathbb{Z}_N^*$，但要满足 $lsb(r)=m$ 的约束；密文是 $c:=[r^e \bmod N]$。参见构造 11.32。

定理 11.33 如果 RSA 问题相对于 GenRSA 是困难的，则构造 11.32 是 **CPA-安全**的。

证明 令 Π 表示构造 11.32。我们证明 Π 在窃听者存在下具有不可区分加密；根据命题 11.3，这蕴含它是 CPA-安全的。

不失一般性，我们可以假设实验 $\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)$ 中的 $m_0=0$ 和 $m_1=1$。因此

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1]=\frac{1}{2}\cdot Pr[\mathcal{A}(N,e,c)=0|c \text{ is an encryption of } 0]+\frac{1}{2}\cdot Pr[\mathcal{A}(N,e,c)=$$

考虑在实验 $\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n)$ 中运行 $\mathcal{A}$。根据定义，

$$Pr[\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n)=1]=Pr[\mathcal{A}(N,e,[r^e \bmod N])=lsb(r)],$$

其中 r 在 $\mathbb{Z}_N^*$ 中是均匀的。由于 $Pr[lsb(r)=1]=1/2$，我们有

$$Pr[\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n)=1]=\frac{1}{2}\cdot Pr[\mathcal{A}(N,e,[r^e \bmod N])=0|lsb(r)=0]+\frac{1}{2}\cdot Pr[\mathcal{A}(N$$

注意到加密 $m\in\{0,1\}$ 恰好对应于选择均匀 r，但要满足 $lsb(r)=m$ 的约束，我们看到

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1]=Pr[\mathrm{RSA}-\mathrm{lsb}_{\mathcal{A},\mathrm{GenRSA}}(n)=1].$$

因此，定理 11.31 蕴含存在一个可忽略函数 negl 使得

$$Pr[\mathrm{PubK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1]\leq\frac{1}{2}+\mathrm{negl}(n),$$

如所期望的。

构造 KEM。 我们现在展示如何扩展构造 11.32 以获得一个**密钥长度为 n 的 KEM**。一个幼稚的方法是简单地选择一个均匀的 n-比特密钥 k，然后使用构造 11.32 的 n 次调用**逐位**加密 k 的比特。这将导致一个相当长的密文，由 n 个 $\mathbb{Z}_N$ 的元素组成。

一个更好的方法是让发送方**重复应用 RSA 排列**（即 e 次方模 N），从一个初始的均匀值 c_1 开始。也就是说，发送方将连续计算 c_1^e，然后是 $(c_1^e)^e=c_1^{e^2}$，依此类推，直到 $c_1^{e^n}$（全部模 N）。最终值 $[c_1^{e^n} \bmod N]$ 将是密文，比特序列 $lsb(c_1),lsb(c_1^e),\ldots,lsb(c_1^{e^{n-1}})$ 是密钥。要解密密文 c，接收方只需**反转**此过程，连续计算 $c^d,(c^d)^d=c^{d^2},\ldots$ 直到 c^{d^n}（再次，全部模 N），以恢复发送方使用的初始值 $c_1=c^{d^n}$。恢复 c_1 后，接收方可以重新计算 $c_1^e,\ldots,c_1^{e^{n-1}}$ 并获得密钥。

可以使用接收方知道群 $\mathbb{Z}_N^*$ 阶的事实，更有效地实现解密。在密钥生成时，接收方可以预先计算 $d':=[d^n \bmod \phi(N)]$ 并将 d' 作为其私钥的一部分存储。然后，接收方可以直接计算 $c_1:=[c^{d'} \bmod N]$，而不是计算 n 次连续模幂运算 $c^d,c^{d^2},\ldots$ 来获得 c_1。这之所以有效，是因为

$$c^{d^n} \bmod N=c^{[d^n \bmod \phi(N)]} \bmod N=c^{d'} \bmod N.$$

上述内容在构造 11.34 中得到了正式描述。

构造 11.34

令 GenRSA 如常，并按如下方式定义 KEM：

- **Gen**：在输入 1^n 时，运行 $\mathrm{GenRSA}(1^n)$ 以获得 (N,e,d)。然后计算 $d':=[d^n \bmod \phi(N)]$（请注意，$\phi(N)$ 可以从 $\langle N,e,d\rangle$ 计算出来，或者在运行 GenRSA 的过程中获得）。输出 $pk=\langle N,e\rangle$ 和 $sk=\langle N,d'\rangle$。

- **Encaps**：在输入 $pk = \langle N, e \rangle$ 和 1^n 时，选择一个均匀的 $c_1 \in \mathbb{Z}_N^*$。然后对于 $i = 1, \ldots, n$：
  i. 计算 $k_i := lsb(c_i)$。
  ii. 计算 $c_{i+1} := [c_i^e \bmod N]$。 输出密文 c_{n+1} 和密钥 $k = k_1 \cdots k_n$。
- **Decaps**：在输入 $sk = \langle N, d' \rangle$ 和密文 c 时，计算 $c_1 := [c^{d'} \bmod N]$。然后对于 $i = 1, \ldots, n$：
  i. 计算 $k_i := lsb(c_i)$。
  ii. 计算 $c_{i+1} := [c_i^e \bmod N]$。 输出密钥 $k = k_1 \cdots k_n$。

使用 RSA 硬核谓词的 KEM。

该构造让人想起 7.4.2 节末尾用于从单向排列构造伪随机生成器的方法。如果我们令 f 表示相对于某个公钥 $\langle N, e \rangle$ 的 RSA 排列（即 $f(x) \triangleq [x^e \bmod N]$），则构造 11.34 的 CPA-安全性等同于 $lsb(f^{n-1}(c_1)), \ldots, lsb(c_1)$ 的**伪随机性**，即使以值 $c = f^n(c_1)$ 为条件。这反过来可以使用定理 11.31 和 7.4.2 节中的技术来证明。（唯一的区别是，在 7.4.2 节中，$f^n(c_1)$ 本身是一个均匀的 n-比特串，而这里它是 $\mathbb{Z}_N^*$ 的一个均匀元素。连续硬核谓词的伪随机性独立于 f 的域。）总结如下：

定理 11.35 如果 RSA 问题相对于 GenRSA 是困难的，则构造 11.34 是 **CPA-安全 KEM**。

效率。 构造 11.34 相当高效。具体来说，假设 $n = 128$，RSA 模数 N 长 2048 位，公指数 e 为 3，因此 e 次幂的模幂运算可以使用两次模乘法计算。（参见附录 B.2.3。）那么加密需要 $2n = 256$ 次模乘法。解密可以通过一次完整的模幂运算（成本约为 $1.5 \cdot 2048 = 3072$ 次模乘法）加上额外的 256 次模乘法来完成。因此，解密成本仅比纯 RSA 加密方案低约 8%。相比之下，加密比纯 RSA 贵得多，但在许多应用中，解密时间更为关键（因为它可能由同时执行数千次解密的服务器实现）。

11.5.4 OAEP 和 RSA PKCS #1 v2.0

到目前为止，我们还没有考虑 **CCA-安全**的基于 RSA 的加密方案。我们首先证明我们迄今为止看到的所有基于 RSA 的加密方案都容易受到**选择密文攻击**。

纯 RSA 加密。 纯 RSA 甚至不是 CPA-安全的。但它确实确保，如果 $m \in \mathbb{Z}_N^*$ 是均匀的，那么窃听 m 相对于公钥 $\langle N, e \rangle$ 的加密 $c = [m^e \bmod N]$ 的攻击者无法恢复 m。即使是这种弱保证，在选择密文攻击可能发生的设置中也不再成立。与 El Gamal 加密的情况一样，这是因为纯 RSA 具有**可塑性**：给定未知消息 m 的加密 $c = [m^e \bmod N]$，很容易生成密文 c'，它是 $[2m \bmod N]$ 的加密，方法是设置

$$c' := [2^e \cdot c \bmod N] = 2^e \cdot m^e = (2m)^e \bmod N.$$

RSA PKCS #1 v1.5。 填充 RSA 加密（在正确的参数设置下被推测为 CPA-安全）容易受到与纯 RSA 加密**本质上相同**的攻击。但对 PKCS #1 v1.5 加密还有一种更有趣的选择密文攻击，与上面提出的攻击相比，它**不需要对解密预言机的完全访问**；它只需要访问一个“部分”解密预言机，该预言机指示解密某个密文是否返回**错误**。这使得攻击更具实用性，因为只要攻击者能够区分接收方在解密成功后的行为与解密失败后的行为（如 3.7.2 节所示的填充预言机攻击），就可以执行该攻击。

回想一下，PKCS #1 v1.5 标准中定义的公钥加密方案使用填充 RSA 加密的一种变体，其中填充以特定方式完成。特别是，填充消息的**两个高阶字节**总是 $0\text{x}00||0\text{x}02$。在解密时，接收方应该检查这两个高阶字节是否与这些值匹配，如果不匹配则返回一个**错误**。1998 年，Bleichenbacher 开发了一种选择密文攻击，该攻击利用了进行此检查的事实。粗略地讲，给定密文 c，它对应于使用公钥 $\langle N, e\rangle$ 对某个未知消息 m 的诚实加密，攻击重复选择均匀 $s \in \mathbb{Z}_N^*$ 并将密文 $c' := [s^e \cdot c \bmod N]$ 提交给接收方。假设 $c = [\hat{m}^e \bmod N]$，其中 $\hat{m} = 0\text{x}00||0\text{x}02||r||0\text{x}00||m$，如 PKCS #1 v1.5 所指定。那么 c' 的解密将给出中间结果 $m' = [s\hat{m} \bmod N]$，并且除非 $\hat{m}'$ 的前两个字节恰好是 $0\text{x}00||0\text{x}02$，否则接收方将返回一个错误。（还会进行其他检查，但为简单起见我们忽略它们。）每当解密成功时，攻击者就会知道 $s\hat{m} \bmod N$ 的前两个字节是 $0\text{x}00||0\text{x}02$，其中 s 是已知的。足够多的此类等式足以让攻击者了解 $\hat{m}$ 并恢复**所有**原始消息 m。

CPA-安全 KEM。 在 11.5.3 节中，我们展示了一个可以被证明基于 RSA 假设是 CPA-安全的 KEM 构造。该构造**也不安全**地抵抗选择密文攻击；我们将细节留作练习。

RSA-OAEP

在本节中，我们探索使用 **最优非对称加密填充 (OAEP)** 构造基于 RSA 的 CCA-安全加密。由此产生的 **RSA-OAEP** 方案遵循以下思想（也在 11.5.2 节中使用）：取消息 m，将其转换为 $\mathbb{Z}_N^*$ 的一个元素 $\hat{m}$，然后令 $c = [\hat{m}^e \bmod N]$ 是密文。然而，这里的转换比以前更复杂。自版本 2.0 以来，RSA-OAEP 的一个版本已作为 **RSA PKCS #1** 的一部分进行了标准化。

令 $l(n), k_0(n), k_1(n)$ 是整数值函数，其中 $k_0(n), k_1(n) = \Theta(n)$，并且 $l(n) + k_0(n) + k_1(n)$ 小于 $\text{GenRSA}(1^n)$ 输出的模数的最小比特长度。固定 n，令 $l = l(n), k_0 = k_0(n), k_1 = k_1(n)$。令 $G : \{0,1\}^{k_0} \to \{0,1\}^{l+k_1}$ 和 $H : \{0,1\}^{l+k_1} \to \{0,1\}^{k_0}$ 是两个哈希函数，它们将被建模为**独立随机预言机**。（尽管在 5.5.1 节中没有讨论使用多个随机预言机，但这以自然的方式解释。）OAEP 定义的转换是一个以 G 和 H 为轮函数的**两轮 Feistel 网络**；参见图 11.4。详细而言，对消息 $m \in \{0,1\}^l$ 的填充如下进行：首先设置 $m' :=$

$m||0^{k_1}$ 并选择一个均匀的 $r \in \{0,1\}^{k_0}$。然后计算

$$s := m' \oplus G(r) \in \{0,1\}^{l+k_1}, \quad t := r \oplus H(s) \in \{0,1\}^{k_0}$$

并设置 $\hat{m} := s||t$。

为了加密消息 m（相对于公钥 $\langle N, e \rangle$），发送方如上生成 $\hat{m}$ 并输出密文 $c := [\hat{m}^e \bmod N]$。（请注意，将 $\hat{m}$ 解释为整数时，由于对 l, k_0, k_1 的限制，它小于 N。）要解密，接收方计算 $\hat{m} := [c^d \bmod N]$ 并令 $s||t := \hat{m}$，其中 s 和 t 具有适当的长度。然后它通过计算 $r := H(s) \oplus t$ 和 $m' := G(r) \oplus s$ 来反转 Feistel 网络。重要的是，接收方随后**验证** m' 的**尾随 k_1 比特是否全为 0**；如果不是，则密文被拒绝并返回一个错误。否则，丢弃 m' 的 k_1 个最低有效 0，并将 m' 的剩余 l 个比特作为消息输出。此过程在构造 11.36 中描述。

RSA-OAEP 的 CCA-安全性证明相当复杂，我们在此不给出。相反，我们仅提供一些直觉。首先考虑

CPA-安全性。 在加密期间，发送方为均匀 r 计算 $m' := m||0^{k_1}, s := m' \oplus G(r), t := r \oplus H(s)$；密文是 $[(s||t)^e \bmod N]$。如果攻击者**从不查询 r 到 G**，那么由于我们将 G 建模为随机函数，从攻击者的角度来看，$G(r)$ 是**均匀**的，因此 m 被一个均匀串掩盖，就像在一次性密码本加密方案中一样。因此，如果攻击者从不查询 r 到 G，则不会泄露有关消息的任何信息。

攻击者可以查询 r 到 G 吗？请注意，r 本身被 $H(s)$ 掩盖。因此，除非攻击者首先查询 s 到 H，否则攻击者不会获得关于 r 的信息。如果攻击者不查询 s 到 H，那么攻击者可能会侥幸猜中 r，但如果我们将 r 的长度（即 k_0）设置得足够长，则此概率是可忽略的。

因此，攻击者了解 m 的唯一方法是**首先查询 s 到 H**。这将要求攻击者从（均匀）密文 $[(s||t)^e \bmod N]$ 中计算 s。请注意，从 $[(s||t)^e \bmod N]$ 计算 s **不是** RSA 问题，RSA 问题涉及同时计算 s 和 t。然而，对于正确的参数设置，我们可以使用定理 11.29 来证明恢复 s 可以在多项式时间内恢复 t，因此如果 RSA 问题是困难的，则恢复 s 是**计算上不可行**的。

论证 CCA-安全性涉及额外的复杂性，但基本思想是证明攻击者进行的每个解密预言机查询 c 都属于以下两类之一：要么攻击者通过**合法加密**某些消息 m 来获得 $\hat{m}$（在这种情况下，攻击者从解密查询中一无所知），要么 c 的解密返回一个**错误**。这是因为接收方在解密期间检查 m' 的 k_1 个低阶比特是否为 0；如果攻击者没有通过合法加密某些消息来构造密文 $\hat{m}$，则此条件成立的概率是可忽略的。形式证明之所以复杂，是因为必须在**不知道私钥**的情况下正确回答攻击者的解密预言机查询，这意味着必须有一种高效的方法来确定是返回错误还是返回消息（如果不返回错误）。这是通过查看对手对随机预言机 G, H 的查询来实现的。

Manger 对 PKCS #1 v2.0 的选择密文攻击。 2001 年，James Manger 展示了对 PKCS #1 v2.0 中指定的 RSA 加密方案的某些实现的选择密文攻击，尽管该规范是 RSA-OAEP 的变体！由于构造 11.36 是 CCA-安全的（假设 RSA 问题是困难的），这怎么可能呢?

检查构造 11.36 中的解密算法，请注意，错误可能以两种方式发生：要么 $\hat{m} \in \mathbb{Z}_N^*$ 太大，要么 $m' \in \{0,1\}^{l+k_1}$ 没有足够的尾随 0。在构造 11.36 中，接收方应该在这两种情况下返回**相同**的错误（表示为 $\perp$）。然而，在某些实现中，接收方会根据哪个步骤失败而输出**不同**的错误。这一个额外的比特信息使攻击者能够发起选择密文攻击，该攻击仅使用大约 $||N||$ 次对泄露错误消息的预言机的查询，即可从消息的加密中恢复**整个消息** m。这表明**严格按照规范实现密码方案的重要性**，因为如果方案的某些方面发生更改，由此产生的证明和分析可能不再适用。

即使在两种情况下返回相同的错误，如果返回错误的时间不同，攻击者也可以确定错误发生在哪里。（这是一个很好的例子，说明攻击者不限于检查算法的输入/输出，而可以使用**侧信道信息**来攻击方案。）实现必须注意确保返回错误的时间是相同的，无论错误发生在哪里。

11.5.5 *随机预言机模型中的 CCA-安全 KEM

我们在此展示了基于 RSA 的 KEM 的一个构造，它在**随机预言机模型**中是 CCA-安全的。（回想一下定理 11.14，任何此类构造都可以与任何 CCA-安全私钥加密方案结合使用，从而提供一个 CCA-安全公钥加密方案。）与上一节中的 RSA-OAEP 方案相比，主要优点是**构造和安全性证明的简单性**。它的主要缺点是，由于它需要 KEM/DEM 范式，因此在加密短消息时会导致**更长的密文**，而 RSA-OAEP 则不需要。然而，对于加密长消息，RSA-OAEP 也会用作混合加密方案的一部分，并且会产生与此处所示的 KEM 获得的加密方案**相似**的效率。

我们描述的 KEM 是 ISO/IEC 18033-2 公钥加密标准的一部分。在该方案中，公钥包括通常的 $\langle N, e\rangle$，并且指定了一个函数 $H : \mathbb{Z}_N^* \to \{0,1\}^n$，该函数将在分析中建模为**随机预言机**。（该函数可以基于某个底层加密哈希函数，如 5.5 节所讨论。我们省略了细节。）为了封装密钥，发送方选择均匀 $r \in \mathbb{Z}_N^*$，然后计算密文 $c := [r^e \bmod N]$ 和密钥 $k := H(r)$。要解密密文 c，接收方只需以通常的方式恢复 r，然后重新派生相同的密钥 $k := H(r)$。参见构造 11.37。

构造 11.37

令 GenRSA 如常，并按如下方式构造 KEM：

- **Gen**：在输入 1^n 时，运行 $\mathrm{GenRSA}(1^n)$ 以计算 (N, e, d)。公钥是 $\langle N, e\rangle$，私钥是 $\langle N, d\rangle$。作为密钥生成的一部分，指定一个函数 $H : \mathbb{Z}_N^* \to \{0,1\}^n$，但我们将其省略。
- **Encaps**：在输入公钥 $\langle N, e\rangle$ 和 1^n 时，选择一个均匀的 $r \in \mathbb{Z}_N^*$。输出密文 $c :=$

$[r^e \bmod N]$ 和密钥 $k := H(r)$。
- **Decaps**：在输入私钥 $\langle N, d\rangle$ 和密文 $c \in \mathbb{Z}_N^*$ 时，计算 $r := [c^d \bmod N]$ 并输出密钥 $k := H(r)$。

随机预言机模型中的 CCA-安全 KEM。

该方案的 CPA-安全性是**直接**的。事实上，密文 c 等于 $[r^e \bmod N]$，其中 $r \in \mathbb{Z}_N^*$ 是均匀的，因此 RSA 假设意味着观察到 c 的窃听者**无法计算** r。这反过来意味着窃听者**不会**查询 r 到 H，因此密钥 $k = H(r)$ 的值从攻击者的角度来看**仍然是均匀的**。

事实上，上述内容也扩展到证明 **CCA-安全性**。这是因为回答任何密文 $\tilde{c} \neq c$ 的解封装预言机查询仅涉及在某个输入 $[\tilde{c}^d \bmod N] = \tilde{r} \neq r$ 处评估 H。因此，攻击者的解封装预言机查询**不会泄露**有关挑战密文封装的密钥 $H(r)$ 的任何额外信息。（形式证明稍微复杂一些，因为我们必须展示如何在**不知道私钥**的情况下模拟解封装预言机查询的答案。然而，事实证明这并不难。）

定理 11.38 如果 RSA 问题相对于 GenRSA 是困难的，并且 H 被建模为**随机预言机**，则构造 11.37 是 **CCA-安全**的。

证明 令 Π 表示构造 11.37，令 $\mathcal{A}$ 是一个概率多项式时间对手。为方便起见，并且因为这是我们第一次使用随机预言机模型的全部功能的证明，我们明确描述实验 $\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$ 的步骤：

1. 运行 $\mathrm{GenRSA}(1^n)$ 以获得 (N, e, d)。此外，选择一个随机函数 $H : \mathbb{Z}_N^* \to \{0,1\}^n$。
2. 选择均匀 $r \in \mathbb{Z}_N^*$，并计算密文 $c := [r^e \bmod N]$ 和密钥 $k := H(r)$。
3. 选择一个均匀比特 $b \in \{0,1\}$。如果 $b = 0$，设置 $\hat{k} := k$。如果 $b = 1$，则选择一个均匀的 $\hat{k} \in \{0,1\}^n$。
4. $\mathcal{A}$ 被给予 $pk = \langle N, e\rangle, c$ 和 $\hat{k}$，并且可以查询 $H(\cdot)$（在任何输入上）和解封装预言机 $Decaps_{\langle N,d\rangle}(\cdot)$（在任何密文 $\tilde{c} \neq c$ 上）。
5. $\mathcal{A}$ 输出一个比特 b'。如果 $b' = b$，则实验的输出定义为 1，否则为 0。

在实验 $\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$ 的一次执行中，令 Query 是 $\mathcal{A}$ 在其执行过程中的任何时刻查询 r 到随机预言机 H 的事件。我们令 $\mathrm{Success}$ 表示事件 $b' = b$（即实验输出 1）。那么

$$Pr[\mathrm{Success}] = Pr[\mathrm{Success} \wedge \overline{\mathrm{Query}}] + Pr[\mathrm{Success} \wedge \mathrm{Query}] \leq Pr[\mathrm{Success} \wedge \overline{\mathrm{Query}}] + Pr[$$

其中所有概率都取自实验 $\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$ 中使用的随机性。我们证明 $Pr[\mathrm{Success} \wedge \overline{\mathrm{Query}}] \leq \frac{1}{2}$ 且 $Pr[\mathrm{Query}]$ 是**可忽略**的。定理成立。

我们首先论证 $Pr[\mathrm{Success} \wedge \overline{\mathrm{Query}}] \leq \frac{1}{2}$。如果 $Pr[\mathrm{Query}] = 0$，这是直接的。否则，

$Pr[\text{Success} \wedge \overline{\text{Query}}] \leq Pr[\text{Success}|\overline{\text{Query}}]$。现在，以 $\overline{\text{Query}}$ 为条件，正确密钥 $k = H(r)$ 的值是均匀的，因为 H 是一个随机函数。考虑 $\mathcal{A}$ 在实验 $\text{KEM}^{\text{cca}}_{\mathcal{A},\Pi}(n)$ 中关于 k 的信息。公钥 pk 和密文 c 本身不包含任何关于 k 的信息。（它们确实唯一地确定了 r，但由于 H 是独立于其他任何事物选择的，因此这没有给出关于 $H(r)$ 的信息。）$\mathcal{A}$ 对 H 进行的查询也不会泄露任何关于 r 的信息，除非 $\mathcal{A}$ 查询 r 到 H（在这种情况下 Query 发生）；这再次依赖于 H 是一个随机函数。最后，$\mathcal{A}$ 对其解封装预言机进行的查询仅泄露 $H(\tilde{r})$，其中 $\tilde{r} \neq r$。这是因为 $Decaps_{\langle N,d\rangle}(\tilde{c}) = H(\tilde{r})$，其中 $\tilde{r} = [\tilde{c}^d \bmod N]$，但 $\tilde{c} \neq c$ 蕴含 $\tilde{r} \neq r$。再次，这个事实以及 H 是一个随机函数的事实意味着，除非 Query 发生，否则不会泄露关于 $H(r)$ 的任何信息。

上述内容表明，只要 Query 不发生，即使给定 $\mathcal{A}$ 对公钥、密文和所有预言机查询答案的视图，正确密钥 k 的值也是**均匀**的。在这种情况下，$\mathcal{A}$ 就无法区分（比随机猜测更好）$\hat{k}$ 是正确密钥还是一个均匀、独立的密钥。因此，$Pr[\text{Success}|\overline{\text{Query}}] = \frac{1}{2}$。

我们强调，在上述论证中，我们**没有**依赖于 $\mathcal{A}$ 是计算有界的这一事实，事实上，即使对 $\mathcal{A}$ 没有施加计算限制，$Pr[\text{Success} \wedge \overline{\text{Query}}] \leq \frac{1}{2}$ 仍然成立。这表明了随机预言机模型的部分威力。

为了完成定理的证明，我们证明：

主张 11.39 如果 RSA 问题相对于 GenRSA 是困难的，并且 H 被建模为**随机预言机**，则 $Pr[\text{Query}]$ 是**可忽略**的。

为了证明这一点，我们构造了一个使用 $\mathcal{A}$ 作为子程序的算法 $\mathcal{A}'$。$\mathcal{A}'$ 被给予 RSA 问题的实例 N, e, c，其目标是计算满足 $r^e \equiv c \bmod N$ 的 r。为此，它将运行 $\mathcal{A}$，回答其对 H 和 $Decaps$ 的查询。

处理对 H 的查询很简单，因为 $\mathcal{A}'$ 可以只返回一个随机值。然而，对 $Decaps$ 的查询比较棘手，因为 $\mathcal{A}'$ **不知道**与有效公钥 $\langle N, e\rangle$ 相关的私钥。

然而，进一步思考，解封装查询也**很容易回答**，因为 $\mathcal{A}'$ 也可以只返回一个**随机值**。也就是说，虽然查询 $Decaps(\tilde{c})$ 应该通过先计算满足 $\tilde{r}^e \equiv \tilde{c} \bmod N$ 的 $\tilde{r}$，然后评估 $H(\tilde{r})$ 来计算，但结果只是一个**均匀值**。因此，$\mathcal{A}'$ 可以简单地返回一个随机值，而无需执行中间计算。唯一的“陷阱”是 $\mathcal{A}'$ 必须确保其对 H-查询和 $Decaps$-查询的答案之间**保持一致性**；即，它必须确保对于任何 $\tilde{r}, \tilde{c}$ 满足 $\tilde{r}^e \equiv \tilde{c} \bmod N$，都有 $H(\tilde{r}) = Decaps(\tilde{c})$。这是通过简单的**簿记**和列表 L_H 和 L_{Decaps} 来处理的，这些列表跟踪 $\mathcal{A}'$ 在响应相应预言机查询时给出的答案。我们现在给出详细信息。

算法 $\mathcal{A}'$：

该算法以 $\langle N, e, c \rangle$ 为输入。

1. 初始化空列表 L_H, L_{Decaps}。选择一个均匀的 $k \in \{0,1\}^n$ 并将 (c, k) 存储在 L_{Decaps} 中。
2. 选择一个均匀比特 $b \in \{0,1\}$。如果 $b = 0$，设置 $\hat{k} := k$。如果 $b = 1$，则选择一个均匀的 $\hat{k} \in \{0,1\}^n$。运行 $\mathcal{A}$（在 N, e, c 和 $\hat{k}$ 上）。

当 $\mathcal{A}$ 进行查询 $H(\tilde{r})$ 时，按如下方式回答：

- 如果 L_H 中存在 $\langle \tilde{r}, k \rangle$ 形式的条目（对于某个 k），则返回 k。
- 否则，令 $\tilde{c} := [\tilde{r}^e \bmod N]$。如果 L_{Decaps} 中存在 $\langle \tilde{c}, k \rangle$ 形式的条目（对于某个 k），则返回 k 并将 $\langle \tilde{r}, k \rangle$ 存储在 L_H 中。
- 否则，选择一个均匀的 $k \in \{0,1\}^n$，返回 k，并将 $\langle \tilde{r}, k \rangle$ 存储在 L_H 中。

当 $\mathcal{A}$ 进行查询 $Decaps(\tilde{c})$ 时，按如下方式回答：

- 如果 L_{Decaps} 中存在 $\langle \tilde{c}, k \rangle$ 形式的条目（对于某个 k），则返回 k。
- 否则，对于 L_H 中的每个条目 $\langle \tilde{r}, k \rangle$，检查 $\tilde{r}^e \equiv \tilde{c} \bmod N$，如果是，则输出 k。
- 否则，选择一个均匀的 $k \in \{0,1\}^n$，返回 k，并将 $\langle \tilde{c}, k \rangle$ 存储在 L_{Decaps} 中。

3. 在 $\mathcal{A}$ 的执行结束时，如果 L_H 中存在 $\langle r, k \rangle$ 形式的条目，使得 $r^e \equiv c \bmod N$，则返回 r。

显然 $\mathcal{A}'$ 在多项式时间内运行，并且 $\mathcal{A}$ 在实验 $\mathrm{RSA} - \mathrm{inv}_{\mathcal{A},\mathrm{GenRSA}}(n)$ 中作为 $\mathcal{A}'$ 的子程序运行时的视图与 $\mathcal{A}$ 在实验 $\mathrm{KEM}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$ 中的视图**相同**：给予 $\mathcal{A}$ 的输入显然具有正确的分布，对 $\mathcal{A}$ 的预言机查询的答案是一致的，并且对所有 H-查询的响应是均匀和独立的。最后，$\mathcal{A}'$ 恰好在 Query 发生时输出正确解。因此，RSA 问题相对于 GenRSA 的困难性蕴含 $Pr[\mathrm{Query}]$ 是**可忽略**的，正如所要求的。

值得指出在上述证明中使用的随机预言机模型（参见 5.5.1 节）的各种属性。首先，我们依赖于以下事实：除非 r 被查询到 H，否则 $H(r)$ 的值是**均匀**的——即使 H 在多个其他值 $\tilde{r} \neq r$ 上被查询。我们还隐含地使用了**可提取性**来论证攻击者不能查询 r 到 H；否则，我们可以使用此攻击者来解决 RSA 问题。最后，证明依赖于**可编程性**，以便模拟对手的解封装预言机查询。

11.5.6 RSA 实现问题和陷阱

我们以简要讨论一些与 **RSA 基方案实现**相关的问题和一些需要注意的**陷阱**来结束本节。

使用中国剩余定理。 在 RSA 基加密的实现中，接收方可以使用**中国剩余定理**（8.1.5 节）来**加速**解密期间模 N 的 e 次根的计算。具体来说，令 $N = pq$，并假设接收方希望使用 $d = [e^{-1} \bmod \phi(N)]$ 计算某个值 y 的 e 次根。接收方可以使用对应关系 $[y^d \bmod N] \leftrightarrow ([y^d \bmod p], [y^d \bmod q])$ 来计算部分结果

$$x_p := [y^d \bmod p] = [y^{[d \bmod (p-1)]} \bmod p] \quad (11.19)$$

和

$$x_q := [y^d \bmod q] = [y^{[d \bmod (q-1)]} \bmod q], \quad (11.20)$$

然后将它们组合以获得 $x \leftrightarrow (x_p, x_q)$，如 8.1.5 节所讨论。请注意，$[d \bmod (p-1)]$ 和 $[d \bmod (q-1)]$ 可以预先计算，因为它们独立于 y。

为什么这更好？假设模一个 l-比特整数的幂运算需要 $\gamma \cdot l^3$ 次操作（对于某个常数 γ）。如果 p, q 各长 n 位，那么朴素地计算 $[y^d \bmod N]$ 需要 $\gamma \cdot (2n)^3 = 8\gamma \cdot n^3$ 步（因为 $||N|| = 2n$）。使用中国剩余定理将其减少到大约 $2 \cdot (\gamma \cdot n^3)$ 步（因为 $||p|| = ||q|| = n$），即大约 $1/4$ 的时间。

示例 11.40

我们回顾示例 8.49。回想一下 $N = 143 = 11 \cdot 13$ 且 $d = 103$，且那里 $y = 64$。为了计算 $[64^{103} \bmod 143]$，我们计算

$$([64 \bmod 11], [64 \bmod 13])^{103} = ([(-2)^{103} \bmod 11], [(-1)^{103} \bmod 13]) = ([(-2)^{[103 \bmod 10]} \text{ mo}$$

$$= ([-8 \bmod 11], -1) = (3, -1).$$

我们可以计算 $1_p = 78 \leftrightarrow (1, 0)$ 和 $1_q = 66 \leftrightarrow (0, 1)$，如 8.1.5 节所讨论。（请注意，这些值可以预先计算，因为它们独立于 y。）然后 $(3, -1) \leftrightarrow 3 \cdot 1_p - 1_q = 3 \cdot 78 - 66 = 168 \equiv 25 \bmod 143$，与先前获得的结果一致。

使用中国剩余定理时的故障攻击。 正如刚刚描述的，在使用中国剩余定理时，应该意识到在计算过程中发生**故障**（或可以被攻击者诱导发生，例如通过硬件篡改）时可能发生的潜在攻击。考虑如果 $[y^d \bmod N]$ 被计算两次时会发生什么：第一次没有错误（给出正确结果 x），但第二次在计算等式 (11.20) 期间出现错误，但等式 (11.19) 没有（相反的情况也适用）。第二次计算产生一个**不正确**的结果 x'，其中 $x' \equiv x \bmod p$ 但 $x' \not\equiv x \bmod q$。这意味着 $p | (x' - x)$ 但 $q \nmid (x' - x)$。但是，那么 $gcd(x' - x, N) = p$，从而得出 N 的因式分解。

一种可能的**对策**是在使用结果之前**验证其正确性**，方法是检查 $x^e \equiv y \bmod N$。（由于

$||e|| \ll ||d||$，使用中国剩余定理仍然提供更高的效率。）建议在硬件实现中这样做。

相关公钥 I。 当多个接收方希望使用相同的加密方案时，他们应该使用**独立的**公钥。此攻击和接下来的攻击证明了当不这样做时可能出错的情况。

想象一家公司希望为其每个员工使用**相同的模数** N。由于不希望加密给一个员工的消息被任何其他员工阅读，该公司为每个员工颁发了不同的 (e_i, d_i) 对。也就是说，第 i 个员工的公钥是 $pk_i = \langle N, e_i \rangle$，他们的私钥是 $sk = \langle N, d_i \rangle$，其中 $e_i \cdot d_i \equiv 1 \bmod \phi(N)$（对于所有 i ）。

这种方法是**不安全**的，并允许任何员工阅读加密给所有其他员工的消息。原因是，正如 8.2.4 节中指出的，给定 N 和 e_i, d_i，且 $e_i \cdot d_i \equiv 1 \bmod \phi(N)$，可以**高效计算 N 的因式分解**。当然，给定 N 的因式分解，就可以计算 $d_j := e_j^{-1} \bmod \phi(N)$（对于任何 j）。

相关公钥 II。 刚刚展示的攻击允许任何员工解密发送给任何其他员工的消息。这仍然留下了以下可能性：只要所有员工相互信任（或者，只要只需要对**外部人员**保密，而不需要对公司内部的其他成员保密），共享模数 N 就可以了。在这里，我们展示了一个场景，表明**共享模数仍然是一个坏主意**，至少在使用纯 RSA 加密时是如此。

假设相同的消息 m 被加密并发送给具有公钥 $\langle N, e_1 \rangle$ 和 $\langle N, e_2 \rangle$ 的两个不同（已知）员工，其中 $e_1 \neq e_2$。进一步假设 $gcd(e_1, e_2) = 1$。那么窃听者看到两个密文

$$c_1 = m^{e_1} \bmod N \quad \text{和} \quad c_2 = m^{e_2} \bmod N.$$

由于 $gcd(e_1, e_2) = 1$，根据命题 8.2，存在整数 X, Y 使得 $Xe_1 + Ye_2 = 1$。此外，给定公指数 e_1 和 e_2，可以使用扩展欧几里得算法（参见附录 B.1.2）高效地计算 X 和 Y。我们声称 $m = [c_1^X \cdot c_2^Y \bmod N]$，可以很容易地计算。这是正确的，因为

$$c_1^X \cdot c_2^Y \equiv (m^{e_1})^X \cdot (m^{e_2})^Y = m^{Xe_1} \cdot m^{Ye_2} = m^{Xe_1 + Ye_2} = m^1 = m \bmod N.$$

如果发送方在加密给两个用户时使用**相同的转换消息** $\hat{m}$，则在使用填充 RSA 或 RSA-OAEP 时也会发生类似的攻击。

RSA 密钥生成中的随机性质量。 在本书中，我们总是假设诚实方可以访问**足够、高质量的随机性**。当违反此假设时，安全性可能会失效。特别是，如果一个 l-比特串是从某个集合 $S \subset \{0,1\}^l$ 中选择的，而不是从 $\{0,1\}^l$ 中均匀选择的，则攻击者可以执行**暴力搜索**（在时间 $\mathcal{O}(|S|)$ 内）来攻击系统。

在某些情况下，情况可能更糟。特别是在 **RSA 密钥生成**的情况下，其中使用一个随机比特样本

r_p 来选择第一个素数 p，使用第二个样本 r_q 来生成第二个素数 q。进一步假设许多公钥/私钥是使用**相同**的**低质量随机性**源生成的，其中 r_p, r_q 是从某个大小为 2^s 的集合 S 中均匀选择的。在生成大约 $2^{s/2}$ 个公钥后（参见附录 A.4），我们期望获得两个不同的模数 N, N'，它们是使用**相同的随机性** $r_p = r'_p$ 生成的。这两个模数共享一个素因子，可以通过计算 $gcd(N, N')$ 轻松找到。因此，攻击者可以从互联网上抓取大量 RSA 公钥，计算它们的**成对最大公约数**，并希望能分解其中的一部分。虽然朴素地计算 $2^{s/2}$ 个模数的成对最大公约数需要时间 $\mathcal{O}(2^s)$，但事实证明，使用一种超出本书范围的“分治”方法，这可以**显着改进**。结果是，攻击者可以在**远低于 2^s 的时间**内执行暴力搜索。此外，即使集合 S 对攻击者未知，该攻击也有效！

上述场景得到了**两个独立工作的研究团队**的实验验证，他们对从互联网上抓取的公钥执行了完全相同的攻击，并成功分解了他们找到的相当一部分密钥。

参考文献和补充阅读

公钥加密的思想最早由 Diffie 和 Hellman 在公开文献中提出 [58]。Rivest、Shamir 和 Adleman [148] 引入了 RSA 假设，并提出了基于此假设的公钥加密方案。正如前一章所指出的，公钥密码学的其他先驱包括 Merkle 和 Rabin（在学术出版物中）以及 Ellis、Cocks 和 Williamson（在机密出版物中）。

定义 11.2 源于 Goldwasser 和 Micali 的开创性工作 [80]，他们也是最早认识到概率加密对于满足此定义的必要性的人。正如第 4 章所指出的，选择密文攻击最初由 Naor 和 Yung [129] 以及 Rackoff 和 Simon [147] 正式定义。Shoup 的解说性文章 [156] 讨论了抵抗选择密文攻击的安全性的重要性。Bellare 等人 [16] 给出了公钥加密的各种安全性概念的统一的现代处理。

混合加密的 CPA-安全性的证明最早由 Blum 和 Goldwasser [36] 给出。CCA-安全性的情况在 [56] 中处理。

令人惊讶的是，El Gamal 加密方案 [70] 直到 1984 年才被提出，尽管它可以被视为 Diffie-Hellman 密钥交换协议的直接转换（参见练习 11.4）。DHIES 在 [2] 中引入。ISO/IEC 18033-2 公钥加密标准可以在 http://www.shoup.net/iso 上找到。

纯 RSA 加密对应于 Rivest、Shamir 和 Adleman [148] 引入的原始方案。11.5.1 节中描述的对纯 RSA 加密的攻击归因于 [161, 55, 84, 47, 40]；参见 [120, 第 8 章] 和 [38] 以获取其他攻击和进一步信息。Coppersmith 定理的证明可以在原始著作 [46] 或随后的几篇阐述中

找到（例如 [119]）。

PKCS #1 RSA 密码学标准（以前和当前版本）可在 http://www.emc.com/emc-plus/rsa-labs 获得。此处描述的对 PKCS #1 v1.5 的选择明文攻击归因于 [49]。Bleichenbacher 对 PKCS #1 v1.5 的选择密文攻击的描述可以在原始论文 [34] 中找到。参见 [12] 以获取后续改进。

定理 11.31 及其推广的证明可以在 [8, 86, 66, 7] 中找到。参见 13.1.2 节以获取此类方案的一般处理。构造 11.37 似乎是由 Shoup [157] 引入并首次分析的。

OAEP 由 Bellare 和 Rogaway [22] 引入。最初的 OAEP 证明后来被发现存在缺陷；感兴趣的读者请参阅 [39, 158, 68]。有关 Manger 对 PKCS #1 v2.0 实现的选择密文攻击的详细信息，请参阅 [117]。

11.5.6 节中描述的成对最大公约数攻击由 Lenstra 等人 [112] 和 Heninger 等人 [88] 进行。

在实践中使用任何加密方案时，都会出现使用哪个密钥长度的问题。不应轻视这个问题，我们请读者参阅 9.3 节及其中的参考文献以进行深入处理。

第一个不依赖于随机预言机模型的 CCA-安全公钥加密方案由 Cramer 和 Shoup [50] 基于 DDH 假设展示。随后，Hoffheinz 和 Kiltz 展示了一个不带随机预言机的基于 RSA 假设的 CCA-安全方案 [92]。