第4章 消息认证码

4.1 消息完整性

4.1.1 保密性对比完整性

密码学最基本的目标之一是使通信双方能够在一个开放的通信信道上**安全地**通信。 但是，“安全通信”究竟意味着什么？ 在第3章中，我们展示了可以在开放信道上实现**秘密通信**。 也就是说，我们展示了如何使用加密来防止窃听者（或可能是更主动的攻击者）获知通过未受保护通信信道发送的消息的任何内容。 然而，并非所有安全问题都与保密性有关。 在许多情况下，保证**消息完整性**（或**消息认证**）的重要性等同或大于保密性，即通信各方应该能够识别其收到的消息是否确实由声称发送该消息的一方发送，并且在传输过程中没有被修改。 我们来看两个典型例子。 考虑一个用户通过互联网与银行通信的情况。 当银行收到将1,000美元从用户账户转账到另一用户 X 账户的请求时，银行必须考虑以下问题：

1. **该请求是否真实可靠？** 也就是说，发出此请求的用户是否是真正的用户，还是由冒充合法用户的攻击者（或许是 X 本人）发出的?
2. 假设转账请求是由合法用户发出的，那么收到的请求细节是否与合法用户的意图完全一致? 或者，例如，转账金额是否被修改了？ 请注意，标准的**错误纠正技术不足以解决第二个问题**。 错误纠正码仅旨在检测和恢复仅影响传输一小部分的**“随机”**错误，但它们对防止恶意攻击者精确选择在何处引入任意数量的错误**无效**。

在实践中出现消息完整性需求的第二个场景与 **Web cookie** 有关。 用于 Web 流量的 HTTP 协议是**无状态的**，因此当客户端和服务器在某个会话中通信时（例如，当 一个用户 [客户端] 在一个商户的 [服务器的] 网站上购物时），作为该会话的一部分生成的任何状态（例如，用户的购物车内容）通常存储在一个 **“cookie”** 中，该 cookie 存储在客户端并作为客户端发送的每条消息的一部分发送到服务器。 假设某个用户存储的 cookie 包含其购物车中的商品及其价格，这可能是因为商户向不同用户提供了不同的价格（例如，反映折扣、促销或用户特定定价）。 在这里，用户应该无法修改他存储的 cookie，以便更改购物车中商品的价格。 因此，商户需要一种技术来确保它存储在用户端的 cookie 的**完整性**。 请注意，cookie 的内容（即商品及其价格）**不是秘密的**，事实上，必须为用户所知。 因此，这里的问题纯粹是**完整性**问题。

一般来说，一个人不能在没有采取特定措施确保通信完整性的情况下**假设**其完整性。 事实上，任何未受保护的在线采购订单、在线银行操作、电子邮件或短信，通常都不能被信任是源自声称的来源，并且在传输过程中没有被修改。 不幸的是，人们通常会信任，因此在许多情况下，像

来电显示或电子邮件返回地址这样的信息被认为是**“来源证明”**，即使它们相对容易伪造。 这为潜在的破坏性攻击敞开了大门。

在本章中，我们将展示如何通过使用密码学技术来防止**未被检测到**的消息篡改，这些消息是通过开放通信信道发送的。 请注意，我们不能指望**完全防止**对消息的恶意篡改，因为那只能在物理层面进行防御。 相反，我们保证**任何此类篡改都将被诚实方检测到**。

4.1.2 加密对比消息认证

正如**保密性**和**消息完整性**的目标不同一样，实现它们的技术和工具也不同。 不幸的是，保密性和完整性经常被混淆和不必要地交织在一起，因此我们首先要明确：**加密（通常）不提供任何完整性**，并且**不应该**在旨在实现消息认证的情况下使用加密，除非它是专门为此目的设计的（我们将在第4.5节中回到这一点）。

人们可能错误地认为加密解决了消息认证的问题。 （事实上，这是一个**常见错误**。） 这是由于模糊且**不正确**的推理，即既然密文完全隐藏了消息的内容，攻击者就不可能以任何有意义的方式修改加密消息。 尽管这种推理具有直观的吸引力，但它完全是**错误的**。 我们通过展示到目前为止我们看到的所有加密方案都**不提供消息完整性**来阐明这一点。

使用流密码加密。 考虑简单的加密方案，其中 $Enc_k(m)$ 计算密文 $c := G(k) \oplus m$，其中 G 是一个伪随机生成器。 在这种情况下，密文非常容易被**操纵**：翻转密文 c 中的任何比特都会导致解密后恢复的消息中的相应比特被翻转。 因此，给定一个加密了（可能未知）消息 m 的密文 c，可以生成一个修改后的密文 c'，使得 $m' := Dec_k(c')$ 与 m 相同，但有一个（或多个）比特被翻转。 这种简单攻击可能导致严重的后果。 作为一个例子，考虑一个用户正在加密他想要从他的银行账户转账的美元金额的情况，其中金额是以二进制表示的。 翻转最低有效位只会将金额更改1美元，但翻转第11个最低有效位会将金额更改超过1,000美元！ （有趣的是，这个例子中的攻击者不一定会知道她是增加还是减少了初始金额，即是翻转了0到1还是反之。 但是如果攻击者对金额有一些**部分知识**——比如一开始小于1,000美元——那么她引入的修改就可以产生**可预测**的效果。） 我们强调，这种攻击**不**与加密方案的**保密性**相矛盾（在定义3.8的意义上）。 事实上，完全相同的攻击适用于**一次性密码本**加密方案，表明即使**完美保密性**也不足以确保**最基本**的消息完整性水平。

使用分组密码加密。 上面描述的攻击利用了密文中的单个比特翻转，除了相应的比特（也被翻转）外，底层明文保持不变的事实。 相同的攻击适用于 OFB 和 CTR 模式加密方案，它们也通过将消息与伪随机流进行异或来加密消息（尽管是每次加密消息时都会更改的流）。 因此，我们看到，即使使用 CPA-安全加密，也不足以防止消息篡改。

人们可能希望攻击 ECB 或 CBC 模式加密会更困难，因为这些情况下的解密涉及**反转**一个

（强）伪随机置换 F，并且我们期望 $F_k^{-1}(x)$ 和 $F_k^{-1}(x')$ 即使 x 和 x' 仅在一个比特上不同，也会**完全不相关**。 （当然，ECB 模式甚至不能保证最基本的保密性概念，但这对于当前的讨论并不重要。） 尽管如此，密文的**单个比特修改**仍然会导致明文的**部分可预测**更改。 例如，当使用 ECB 模式时，翻转密文的第 i 个块中的一个比特**仅**影响明文的第 i 个块——所有其他块保持不变。 尽管对明文的第 i 个块的影响可能无法预测，但更改该块（同时保持其他所有块不变）可能代表一种有害的攻击。 此外，可以通过简单地更改相应密文块的顺序来**更改明文块的顺序**（而不会破坏任何块），并且可以通过简单地丢弃密文块来**截断消息**。

类似地，当使用 CBC 模式时，翻转 IV 的第 j 个比特**仅**更改第一条消息块 m_1 的第 j 个比特（因为 $m_1 := F_k^{-1}(c_1) \oplus IV'$，其中 IV' 是修改后的 IV）； 除了第一个块之外，所有明文块都保持不变（因为第 i 个明文块是计算为 $m_i := F_k^{-1}(c_i) \oplus c_{i-1}$，而块 c_i 和 c_{i-1} 没有被修改）。 因此，CBC 加密的**第一块消息**可以被**任意更改**。 这在实践中是一个严重的问题，因为第一块通常包含重要的**头部信息**。

最后，请注意，到目前为止我们看到的所有加密方案都具有这样的属性：**每个密文**（可能满足某些长度约束）都对应于某个**有效消息**。 因此，即使攻击者不知道底层消息是什么，她也可以通过发送某个**任意密文**来“欺骗”通信方之一——这使得**攻击者很容易冒充**。 正如我们将在第4.5节中正式定义**认证加密**时看到的那样，即使是这种类型的攻击也应该被排除。

4.2 消息认证码 - 定义

我们已经看到，一般来说，加密并不能解决消息完整性问题。相反，需要一个额外的机制，使通信双方能够知道消息是否被篡改。完成这项任务的正确工具是**消息认证码 (Message Authentication Code, MAC)**。

消息认证码的目标是防止攻击者在一条消息从一方发送到另一方时对其进行修改，或注入一条新消息，而接收方却未检测到该消息并非源自预定方。与加密的情况一样，这只有在通信双方共享了攻击者不知道的某个秘密时才有可能（否则，攻击者可以冒充发送方发送消息，从而无法阻止）。在这里，我们将继续考虑**私钥设置**，即双方共享相同的秘密密钥。

消息认证码的语法

在正式定义消息认证码的安全性之前，我们首先定义 MAC 是什么以及如何使用它。希望以认证方式通信的两个用户首先通过生成并**共享**一个秘密密钥 k 来开始通信。当一方想要发送消息 m 给另一方时，她根据该消息和共享密钥计算一个 MAC 标签（或简称**标签**）t，并将消息 m 和标签 t 发送给另一方。用我们前面说的话来重述，消息 m 的发送方计算 $t \leftarrow \mathrm{Mac}_k(m)$ 并将 (m, t) 传输给接收方。接收方使用共享密钥以及消息 m 和标签 t 作为输入来运行**验证算法**

Vrfy，并指示给定的标签是否有效。形式上：

定义 4.1 消息认证码（或 MAC）由三个概率多项式时间算法 $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ 组成，满足：

1. **密钥生成算法** Gen 以安全参数 1^n 作为输入，输出一个密钥 k，满足 $|k| \geq n$。
2. **标签生成算法** Mac 以密钥 k 和消息 $m \in \{0,1\}^*$ 作为输入，输出一个标签 t。由于该算法可以是随机化的，我们将其记为 $t \leftarrow \mathrm{Mac}_k(m)$。
3. **确定性验证算法** Vrfy 以密钥 k、消息 m 和标签 t 作为输入。它输出一个比特 b，其中 $b = 1$ 表示有效，$b = 0$ 表示无效。我们将其记为 $b := \mathrm{Vrfy}_k(m, t)$。

要求对于每个 n、每个由 $\mathrm{Gen}(1^n)$ 输出的密钥 k 以及每条消息 $m \in \{0,1\}^*$，都有 $\mathrm{Vrfy}_k(m, \mathrm{Mac}_k(m)) = 1$。

如果存在一个函数 ℓ 使得对于每个由 $\mathrm{Gen}(1^n)$ 输出的 k，算法 Mac_k 仅对长度为 $\ell(n)$ 的消息 $m \in \{0,1\}^{\ell(n)}$ 定义，则我们称该方案是**用于长度为 $\ell(n)$ 消息的固定长度 MAC**。

与私钥加密一样，$\mathrm{Gen}(1^n)$ 几乎总是简单地选择一个均匀的密钥 $k \in \{0,1\}^n$，在这种情况下我们省略 Gen。

规范验证 (Canonical verification)。 对于确定性消息认证码（即 Mac 是确定性算法的情况），执行验证的规范方法是简单地重新计算标签并检查是否相等。换句话说，$\mathrm{Vrfy}_k(m, t)$ 首先计算 $\tilde{t} := \mathrm{Mac}_k(m)$，然后当且仅当 $\tilde{t} = t$ 时输出 1。然而，即使对于确定性 MAC，定义一个单独的 Vrfy 算法也是有用的，以便明确区分认证消息的语义与验证其真实性的语义。

消息认证码的安全性

我们现在定义消息认证码的默认安全概念。该定义的直观思想是：没有高效的攻击者能够对任何**以前未曾**发送（和认证）过的“新”消息生成有效标签。

与任何安全定义一样，为了形式化这个概念，我们必须定义攻击者的能力以及什么应被视为“破解”。像往常一样，我们只考虑概率多项式时间攻击者，因此真正的问题在于我们如何对攻击者与通信双方的交互进行建模。

在消息认证码的设置中，观察通信的攻击者可能能够看到这些方发送的所有消息及其对应的 MAC 标签。攻击者也可能能够影响这些消息的内容，无论是直接还是间接的（例如，如果攻击者的外部行为影响了双方发送的消息，则为间接影响）。例如，在前面提到的网络 cookie 示例中，用户自己的行为影响了存储在他计算机上的 cookie 的内容，这是真实的。

为了正式地对上述情况建模，我们允许攻击者请求对其选择的任何消息进行 MAC 标签。形式上，我们赋予攻击者对 $\mathrm{Mac}_k(\cdot)$ 的 MAC 预言机访问权限；攻击者可以重复地向该预言机提交其选择的任何消息 m，并返回一个标签 $t \leftarrow \mathrm{Mac}_k(m)$。（对于固定长度 MAC，只能提交正确长度的消息。）

如果攻击者能够输出任何消息 m 及其标签 t，使得：(1) t 是消息 m 的有效标签（即 $\mathrm{Vrfy}_k(m,t)=1$），并且 (2) 攻击者**以前没有**请求过消息 m 的 MAC 标签（即，来自其预言机的），则我们认为这是对方案的“破解”。第一个条件意味着，如果攻击者将 (m,t) 发送给诚实的一方，那么该方将错误地认为 m 源自合法方，因为 $\mathrm{Vrfy}_k(m,t)=1$。第二个条件是必需的，因为攻击者总是可以简单地复制合法方先前发送的消息和 MAC 标签（当然，这些复制的消息和标签将被接受为有效）。这种**重放攻击 (replay attack)** 不被认为是消息认证码的“破解”。这并不意味着重放攻击不是一个安全问题；它们是，我们将在下面对此进行更多讨论。

满足上述安全级别的 MAC 被称为**在适应性选择消息攻击下存在不可伪造性 (existentially unforgeable under an adaptive chosen-message attack)**。“存在不可伪造性”指的是攻击者不能对任何消息伪造出有效标签，“适应性选择消息攻击”指的是攻击者能够在攻击过程中适应性地选择任意消息以获取 MAC 标签。

为了形式化定义，考虑消息认证码 $\Pi=(\mathrm{Gen},\mathrm{Mac},\mathrm{Vrfy})$、攻击者 $\mathcal{A}$ 和安全参数 n 的以下实验：

消息认证实验 $\mathrm{Mac\text{-}forge}_{\mathcal{A},\Pi}(n)$：

1. 运行 $\mathrm{Gen}(1^n)$ 生成密钥 k。
2. 攻击者 $\mathcal{A}$ 获得输入 1^n 和对 $\mathrm{Mac}_k(\cdot)$ 的预言机访问权限。攻击者最终输出 (m,t)。设 $\mathcal{Q}$ 表示 $\mathcal{A}$ 向其预言机询问过的所有查询的集合。
3. 当且仅当 (1) $\mathrm{Vrfy}_k(m,t)=1$ 且 (2) $m\notin\mathcal{Q}$ 时，$\mathcal{A}$ 成功。在这种情况下，实验的输出定义为 1。

如果任何高效攻击者在上述实验中成功的概率可忽略，则 MAC 是安全的：

定义 4.2 消息认证码 $\Pi=(\mathrm{Gen},\mathrm{Mac},\mathrm{Vrfy})$ 在适应性选择消息攻击下是**存在不可伪造**的，或简称为**安全**的，如果对于所有概率多项式时间攻击者 $\mathcal{A}$，存在一个可忽略函数 negl，使得

$$\Pr[\mathrm{Mac\text{-}forge}_{\mathcal{A},\Pi}(n)=1]\le\mathrm{negl}(n).$$

这个定义是否过于严格？ 上述定义在两个方面相当严格。首先，攻击者可以请求对其选择的**任何**消息进行 MAC 标签。其次，如果攻击者能够对**任何**以前未经认证的消息输出一个有效标签，则该方案被认为是“破解”了。有人可能会反对，这两个组成部分都不切实际且过于严格：在“现

实世界”中使用的 MAC 中，诚实方只会认证“有意义”的消息（攻击者可能只对此有有限的控制），同样，只有当攻击者能够伪造一个“有意义”的消息的有效标签时，才应被视为安全漏洞。为什么不调整定义来捕捉这一点呢？

关键点在于，什么构成*有意义*的消息完全取决于应用程序。虽然 MAC 的某些应用可能只认证英文文本消息，其他应用可能认证电子表格文件、数据库条目，还有些可能认证原始数据。协议也可能被设计成认证任何内容——事实上，某些用于实体认证的协议正是如此。通过使 MAC 的安全定义尽可能强大，我们确保安全的 MAC 广泛适用于各种目的，而不必担心 MAC 与应用程序语义的兼容性。

重放攻击。 我们强调，上述定义以及消息认证码本身并不能防止**重放攻击**，即攻击者将先前发送的消息（及其 MAC 标签）重放给诚实的一方。然而，重放攻击是一个严重的问题！再次考虑用户（例如，爱丽丝）向她的银行发送请求，要求从她的账户中转账 1,000 美元给另一个用户（例如，鲍勃）的场景。爱丽丝在发送时会计算一个 MAC 标签并将其附加到请求上，这样银行就知道该请求是真实的。如果 MAC 是安全的，鲍勃将无法拦截请求并将金额改为 10,000 美元，因为这将涉及对以前未经认证的消息伪造有效标签。然而，没有什么可以阻止鲍勃拦截爱丽丝的消息并将其**重放**给银行十次。如果银行接受了这些消息中的每一个，那么净效果是 10,000 美元将转入鲍勃的账户，而不是预期的 1,000 美元。

尽管重放攻击构成了真正的威胁，但 MAC 本身无法防御此类攻击，因为 MAC 的定义（定义 4.1）并未将任何状态概念纳入验证算法（因此每次向验证算法提交有效的 (m, t) 对时，它都会始终输出 1）。相反，如果需要此类保护，则必须由某些更高级别的应用程序来处理对重放攻击的保护。消息认证码的定义以这种方式构建的原因是，我们再次不愿对使用 MAC 的应用程序做任何语义假设；特别是，关于重放消息是否应被视为“有效”的决定可能是依赖于应用程序的。

防止重放攻击的两种常用技术是使用**序列号**（也称为**计数器**）或**时间戳**。第一种方法（在 4.5.3 节中更一般的上下文中描述）要求通信用户保持（同步的）状态，并且当用户通过偶尔丢失消息的有损信道通信时，可能会出现问题（尽管这个问题可以缓解）。在第二种方法中，使用时间戳，发送方在认证消息之前将当前时间 T（例如，到最近的毫秒）附加到消息上，并与结果标签 t 一起发送 T、消息 m。当接收方收到 T, m, t 时，它验证 t 是 $T||m$ 的有效标签，并且 T 在接收方当前时间 T' 的某个可接受的时钟偏差范围内。这种方法也有一些缺点，包括发送方和接收方需要保持紧密同步的时钟，以及如果重放攻击进行得足够快（特别是在可接受的时间窗口内），重放攻击仍有可能发生。

强 MACs。 根据定义，安全的 MAC 确保攻击者无法对以前从未认证过的消息生成有效标签。但它并没有排除攻击者可能能够对以前认证过的消息生成**不同**标签的可能性。也就是说，MAC

保证如果攻击者获得了消息 $m_1, \ldots$ 上的标签 $t_1, \ldots$，那么它将无法对任何消息 $m \notin \{m_1, \ldots\}$ 伪造有效标签 t。然而，攻击者可能有可能对消息 m_1“伪造”一个不同的有效标签 $t_1' \neq t_1$。一般来说，这种类型的攻击行为并不是一个安全问题。尽管如此，在某些设置中，考虑一个**更强**的 MAC 安全定义是有用的，它排除了这种行为。

形式上，我们考虑一个修改后的实验 $\mathrm{Mac\text{-}sforge}$，它的定义与 $\mathrm{Mac\text{-}forge}$ 完全相同，除了现在集合 $\mathcal{Q}$ 包含预言机查询及其关联响应对。（也就是说，如果 $\mathcal{A}$ 查询 $\mathrm{Mac}_k(m)$ 并收到响应标签 t，则 $(m, t) \in \mathcal{Q}$。）当且仅当 (1) $\mathrm{Vrfy}_k(m, t) = 1$ 且 (2) $(m, t) \notin \mathcal{Q}$ 时，攻击者 $\mathcal{A}$ 成功（并且实验 $\mathrm{Mac\text{-}sforge}$ 的评估结果为 1）。

定义 4.3 消息认证码 $\Pi = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ 是**强安全**的，或**强 MAC**，如果对于所有概率多项式时间攻击者 $\mathcal{A}$，存在一个可忽略函数 negl，使得

$$\Pr[\mathrm{Mac\text{-}sforge}_{\mathcal{A},\Pi}(n) = 1] \leq \mathrm{negl}(n).$$

不难看出，如果一个安全的 MAC 使用规范验证，那么它也是强安全的。这一点很重要，因为所有现实世界中的 MAC 都使用规范验证。我们把以下命题的证明留作练习。

命题 4.4 设 $\Pi = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ 是一个使用规范验证的安全 MAC。那么 Π 是一个强 MAC。

验证查询 (Verification Queries)

定义 4.2 和 4.3 赋予攻击者对 MAC 预言机的访问权限，这对应于一个可以影响诚实发送方为某些消息 m 生成标签的现实世界攻击者。我们也可以考虑一个与诚实接收方交互的攻击者，它向接收方发送 m', t' 以了解 $\mathrm{Vrfy}_k(m', t') = 1$ 是否成立。这样一个攻击者可以用一种自然的方式在上述定义中形式化，即也赋予攻击者对验证预言机的访问权限。

以这种方式将验证预言机纳入定义，也许是定义消息认证码安全性的“正确”方式。然而，事实证明，对于使用规范验证的 MAC，这并**没有区别**：任何满足定义 4.2 的 MAC 也满足允许验证查询的定义变体。同样，任何强 MAC 也会自动保持安全，即使在允许验证查询的情况下也是如此。（事实上，这正是定义强 MAC 安全性的动机之一。）然而，对于不使用规范验证的 MAC，允许验证查询可能会产生差异；请参阅练习 4.2 和 4.3。由于本书涵盖的大多数 MAC（以及实践中使用的 MAC）都使用规范验证，因此我们使用省略对验证预言机访问权限的传统定义。

潜在的计时攻击 (A potential timing attack)。 上述内容中未解决的一个问题是，可能对 MAC 验证进行**计时攻击**。在这里，我们考虑一个攻击者，他可以向接收方发送消息/标签对——从而使用接收方作为验证预言机——并且不仅了解接收方是否接受或拒绝，而且还了解接收方做

出此决定所需的时间。我们证明，如果这种攻击是可能的，那么 MAC 验证的自然实现会导致容易被利用的漏洞。

（请注意，在我们通常的密码学安全定义中，攻击者只了解其有权访问的预言机的输出，而不了解任何其他信息。我们在此描述的攻击是**侧信道攻击**的一个例子，它表明某些现实世界中的攻击未被通常的定义所捕获。）

具体来说，假设 MAC 使用规范验证。为了验证消息 m 上的标签 t，接收方计算 $t' := \mathrm{Mac}_k(m)$，然后将 t' 与 t 进行比较，当且仅当 t' 和 t 相等时输出 1。假设此比较是使用标准例程（例如 C 语言中的 strcmp）实现的，该例程一次比较一个字节，并且在遇到第一个不相等的字节时立即拒绝。由此观察到的结果是，拒绝所需的时间取决于第一个不相等字节的**位置**。

可以使用这种看似微不足道的信息来伪造任何所需消息 m 的标签。基本思想是：假设攻击者知道消息 m 的正确标签的前 i 个字节。（开始时，$i = 0$。）攻击者将通过发送 $(m, t_0), \ldots, (m, t_{255})$ 给接收方来了解正确标签的下一个字节，其中 t_j 是一个字符串，其前 i 个字节设置正确，第 $(i+1)$ 个字节等于 j（十六进制），其余字节设置为 $0\mathrm{x}00$。所有这些标签很可能会被拒绝（如果不是，那么攻击者无论如何都会成功）；然而，对于其中恰好一个标签，前 $(i+1)$ 个字节将与正确标签匹配，拒绝所需的时间会比其余标签稍长。如果 t_j 是导致拒绝时间最长的标签，则攻击者了解到正确标签的第 $(i+1)$ 个字节是 j。通过这种方式，攻击者只需最多 256 个验证预言机查询即可了解正确标签的每个字节。对于一个 16 字节的标签，这种攻击在最坏情况下只需要 4096 次查询。

有人可能会怀疑这种攻击是否现实，因为它需要对验证预言机的访问权限以及测量比较 i 个字节与比较 $i+1$ 个字节所需时间差异的能力。事实上，正是针对真实系统进行了这种攻击！仅举一个例子，MAC 用于验证 $\mathrm{Xbox}\ 360$ 中的代码更新，其中 MAC 验证的实现存在 2.2 毫秒的拒绝时间差异。攻击者能够利用这一点并将盗版游戏加载到硬件上。

基于上述情况，我们得出结论，MAC 验证应该使用**与时间无关**的字符串比较，即始终比较所有字节。

4.3 构建安全消息认证码

4.3.1 一个定长 MAC

伪随机函数是构造安全消息认证码的自然工具。 直觉是，如果 **MAC 标签** t 是通过对消息 m 应用**伪随机函数**获得的，那么在**以前未认证**的消息上伪造标签需要攻击者正确猜测**“新”输入点**上**伪随机函数的值。 猜测**随机函数**在新点上的值的概率是 2^{-n}（如果函数的输出长度是 n）。 猜测**伪随机函数**的这样一个值的概率只能是**可忽略地更大**。

如**构造 4.5** 所示，上述想法适用于构造一个用于加密长度为 n 的消息的**安全定长 MAC**（因为我们的伪随机函数默认具有 n 比特的分组长度）。 这很有用，但离我们的目标还差得远。 在第4.3.2节中，我们将展示如何扩展它以处理**任意长度**的消息。 我们将在第4.4节和5.3.2节中探讨更高效的任意长度消息 MAC 构造。

构造 4.5 令 F 是一个伪随机函数。 定义一个用于加密长度为 n 的消息的**定长 MAC** 如下： • **Mac：** 输入密钥 $k \in \{0,1\}^n$ 和消息 $m \in \{0,1\}^n$，输出标签 $t := F_k(m)$。 （如果 $|m| \neq |k|$，则不输出任何内容。） • **Vrfy：** 输入密钥 $k \in \{0,1\}^n$、消息 $m \in \{0,1\}^n$ 和标签 $t \in \{0,1\}^n$，当且仅当 $t = F_k(m)$ 时输出1。 （如果 $|m| \neq |k|$，则输出0。） **一个从任何伪随机函数构造的定长 MAC。**

定理 4.6 如果 F 是一个伪随机函数，则**构造 4.5** 是一个用于加密长度为 n 的消息的**安全定长 MAC**。

证明 与以前使用伪随机函数一样，该证明遵循**首先分析使用真正随机函数**的方案的安全性，然后考虑用伪随机函数替换真正随机函数的结果的范式。

令 A 是一个任意概率多项式时间攻击者。 考虑消息认证码 $\overline{\Pi} = (\overline{Gen}, \overline{Mac}, \overline{Vrfy})$，它与**构造 4.5** 中的 $\Pi = (Mac, Vrfy)$ **相同**，除了使用**真正随机函数** f 代替伪随机函数 F_k。 也就是说， $\overline{Gen}(1^n)$ 通过选择一个**均匀函数** $f \in Func_n$ 来工作，而 $\overline{Mac}$ 像 Mac 一样计算标签，只是使用 f 代替 F_k。 可以立即得出

$$\Pr[Mac\text{-}forge_{A,\overline{\Pi}}(n) = 1] \leq 2^{-n} \tag{4.1}$$

因为对于任何消息 $m \notin Q$，从攻击者的角度来看，值 $t = f(m)$ 在 $\{0,1\}^n$ 中是均匀分布的。

我们接下来证明存在一个**可忽略函数** $negl$ 使得

$$\left|\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1] - \Pr[Mac\text{-}forge_{A,\overline{\Pi}}(n) = 1]\right| \leq negl(n); \tag{4.2}$$

结合公式(4.1)，这表明

$$\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1] \leq 2^{-n} + negl(n),$$

从而证明了定理。

为了证明公式(4.2)，我们构造一个**多项式时间区分器** D，它被赋予对某个函数的预言机访问权限，其目标是确定这个函数是伪随机的（即等于均匀 $k \in \{0,1\}^n$ 的 F_k）还是随机的（即等于均匀 $f \in Func_n$ 的 f）。 为此， D 模拟 A 的消息认证实验，并观察 A 是否成功地在

“新”消息上输出了一个有效标签。 如果是，则 D 猜测其预言机是一个伪随机函数；否则， D 猜测其预言机是一个随机函数。 详细地说：

区分器 D： D 获得输入 1^n 和对预言机 $O : \{0,1\}^n \to \{0,1\}^n$ 的访问权限，并按以下方式工作：

1. 运行 $A(1^n)$。 每当 A 查询其 MAC 预言机关于消息 m 时（即，每当 A 请求消息 m 上的标签时），以以下方式回答此查询： 查询 O 关于 m 并获得响应 t；将 t 返回给 A。
2. 当 A 在执行结束时输出 $\langle m, t \rangle$ 时，执行以下操作： (a) 查询 O 关于 m 并获得响应 $\hat{t}$。 (b) 如果 (1) $\hat{t} = t$ 并且 (2) A 从未查询其 MAC 预言机关于 m，则输出1；否则，输出0。

显然 D 在多项式时间内运行。

请注意，如果 D 的预言机是一个伪随机函数，那么 A 作为子程序在 D 运行时看到的视图与 A 在实验 $Mac\text{-}forge_{A,\Pi}(n)$ 中看到的视图**分布相同**。 此外， D 恰好在 $Mac\text{-}forge_{A,\Pi}(n) = 1$ 时输出1。 因此

$$\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] = \Pr[Mac\text{-}forge_{A,\Pi}(n) = 1],$$

其中 $k \in \{0,1\}^n$ 是在上述中均匀选择的。 如果 D 的预言机是一个随机函数，那么 A 作为子程序在 D 运行时看到的视图与 A 在实验 $Mac\text{-}forge_{A,\bar{\Pi}}(n)$ 中看到的视图**分布相同**，并且 D 再次恰好在 $Mac\text{-}forge_{A,\bar{\Pi}}(n) = 1$ 时输出1。 因此

$$\Pr\left[D^{f(\cdot)}(1^n) = 1\right] = \Pr[Mac\text{-}forge_{A,\bar{\Pi}}(n) = 1],$$

其中 $f \in Func_n$ 是均匀选择的。

由于 F 是一个伪随机函数，并且 D 在多项式时间内运行，存在一个**可忽略函数** $negl$ 使得

$$\left|\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] - \Pr\left[D^{f(\cdot)}(1^n) = 1\right]\right| \leq negl(n).$$

这蕴含了公式(4.2)，完成了定理的证明。

4.3.2 MAC 的域扩展

构造 4.5 是重要的，因为它展示了**从伪随机函数构造安全消息认证码**的一般范式。 不幸的是，该构造仅能处理**定长**消息，而且长度相当**短**。 在大多数应用中，这些限制是**不可接受**的。 我们在此展示如何从任何**定长**的、用于加密长度为 n 的消息的 MAC 构造一个处理**任意长度**消息的

通用 MAC。 我们展示的构造**效率不高**，在实践中不太可能被使用。 确实，已知有**更高效**的安全 MAC 构造，如我们在第4.4节和5.3.2节中讨论的那样。 我们包含当前的构造是为了其**简单性**和**一般性**。

令 $\Pi' = (Mac', Vrfy')$ 是一个用于加密长度为 n 的消息的**安全定长 MAC**。 在介绍基于 Π' 的**任意长度消息 MAC** 的构造之前，我们排除一些简单的想法，并描述一些**必须防止**的规范攻击。 下面，我们将要认证的消息 m 解析为一系列块 $m_1, \ldots, m_d$；请注意，由于我们的目标是处理**任意长度**的消息， d 可以从消息到消息变化。

1. 一个自然的第一个想法是简单地**单独认证每个块**，即计算 $t_i := Mac'_k(m_i)$ 对于所有 i，并输出 $t_1, \ldots, t_d$ 作为标签。 这防止了攻击者在未被检测到的情况下发送任何以前未认证的块。 然而，这**不**防止**块重排序攻击**，其中攻击者打乱认证消息中块的顺序。 具体来说，如果 $\langle t_1, t_2 \rangle$ 是消息 m_1, m_2（其中 $m_1 \neq m_2$）上的有效标签，那么 $\langle t_2, t_1 \rangle$ 是消息 m_2, m_1（不同的）上的有效标签（定义4.2不允许这种情况）。
2. 我们可以通过** along with each block** 认证**块索引**来防止前一次攻击。 也就是说，我们现在计算 $t_i = Mac'_k(\langle i, m_i \rangle)$ 对于所有 i，并输出 $\langle t_1, \ldots, t_d \rangle$ 作为标签。 （请注意，块长度 $|m_i|$ 将需要更改。） 这**不**防止**截断攻击**，其中攻击者简单地丢弃消息末尾的块（并丢弃相应的标签块）。
3. 截断攻击可以通过**额外认证消息长度**以及每个块来阻止。 （**单独**认证消息长度作为单独的块是**不**起作用的。 您知道为什么吗？） 也就是说，计算 $t_i = Mac'_k(\langle l, i, m_i \rangle)$ 对于所有 i，其中 l 表示消息的长度（以比特为单位）。 （再次，块长度 $|m_i|$ 将需要减小。） 该方案容易受到**“混合匹配”攻击**，其中攻击者将来自**不同消息**的块组合在一起。 例如，如果攻击者获得了消息 $m = m_1, \ldots, m_d$ 和 $m' = m'_1, \ldots, m'_d$ 上的标签 $\langle t_1, \ldots, t_d \rangle$ 和 $\langle t'_1, \ldots, t'_d \rangle$，她可以输出消息 $m_1, m'_2, m_3, m'_4, \ldots$ 上的有效标签 $t_1, t'_2, t_3, t'_4, \ldots$。

我们可以通过在每个块中包含一个**随机“消息标识符”来防止最后一次攻击，该标识符可防止来自不同消息的块组合在一起。 这导致了构造 4.7**。

构造 4.7 令 $\Pi' = (Mac', Vrfy')$ 是一个用于加密长度为 n 的消息的**定长 MAC**。 定义一个 MAC Π 如下： • **Mac：** 输入密钥 $k \in \{0,1\}^n$ 和**（非零）长度 $l < 2^{n/4}$** 的消息 $m \in \{0,1\}^*$，将 m 解析为 d 个块 $m_1, \ldots, m_d$，每个块长度为 $n/4$。 （如果需要，最后一个块用0填充。） 选择一个**均匀标识符** $r \in \{0,1\}^{n/4}$。 对于 $i = 1, \ldots, d$，计算 $t_i \leftarrow Mac'_k(\langle r, l, i, m_i \rangle)$，其中 i, l 被编码为长度为 $n/4$ 的字符串。 输出标签 $t := \langle r, t_1, \ldots, t_d \rangle$。 • **Vrfy：** 输入密钥 $k \in \{0,1\}^n$、长度 $l < 2^{n/4}$ 的消息 $m \in \{0,1\}^*$ 和标签 $t = \langle r, t_1, \ldots, t_d \rangle$，将 m 解析为 d' 个块 $m'_1, \ldots, m'_{d'}$，每个块长度为 $n/4$。 （如果需要，最后一个块用0填充。） 当且仅当 $d' = d$ 并且对于 $1 \leq i \leq d$ 都有 $Vrfy'_k(\langle r, l, i, m'_i \rangle, t_i) = 1$ 时，输出1。

† 注意 i 和 l 可以用 $n/4$ 比特编码，因为 $i, l < 2^{n/4}$。 **一个从任何定长 MAC 构造的任意长度消息 MAC。**

（技术上，该方案仅处理长度**小于** $2^{n/4}$ 的消息。 渐近地，由于这是一个指数界限，诚实方不会认证这么长的消息，并且任何多项式时间攻击者都不能向其 MAC 预言机提交这么长的消息。在实践中，当 n 的一个具体值被固定时，必须确保该界限是可以接受的。）

定理 4.8 如果 Π' 是一个用于加密长度为 n 的消息的**安全定长 MAC**，则**构造 4.7** 是一个**安全 MAC**（用于**任意长度**消息）。

证明 直觉是，只要 Π' 是安全的，攻击者就不能引入带有有效标签的新块。 此外，每个块中包含的额外信息防止了前面概述的各种攻击（丢弃块、重排序块等）。 我们将通过展示这些攻击是唯一可能的攻击来证明安全性。

令 Π 是由**构造 4.7** 给出的 MAC，并令 A 是一个概率多项式时间攻击者。 我们证明 $\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1]$ 是**可忽略的**。 我们首先引入一些将在证明中使用的符号。 令 $Repeat$ 表示在实验 $Mac\text{-}forge_{A,\Pi}(n)$ 中 MAC 预言机返回的标签中有**相同的随机标识符**出现的事件。 令 $\langle m, t = \langle r, t_1, \ldots \rangle \rangle$ 表示 A 的最终输出，其中 $m = m_1, \ldots$ 的长度为 l，我们令 $NewBlock$ 是**至少一个块 $\langle r, l, i, m_i \rangle$ 以前从未在回答 A 的 Mac 查询过程中被 Π' 认证的事件**。 （请注意，根据 Π 的构造，在计算 $Mac_k(m)$ 时，很容易判断哪些块被 Mac'_k 认证了。） 非正式地说， $NewBlock$ 是 A 尝试在底层定长 MAC Π' 以前从未认证过的块上输出一个有效标签的事件。 我们有

$$\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1] = \Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge Repeat] + \Pr[Mac\text{-}forge_{A,\Pi}(n) =$$

$$\leq \Pr[Repeat] + \Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge \overline{Repeat}]$$

$$\leq \Pr[Repeat] + \Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge \overline{Repeat} \wedge NewBlock] + \Pr[Mac\text{-}forge_{A,\Pi}(n) \quad (4.3)$$

我们证明公式(4.3)右侧的前两项是**可忽略的**，最后一项是0。 这蕴含着 $\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1]$ 是可忽略的，正如所期望的。

断言 4.9 $\Pr[Repeat]$ 是可忽略的。

证明 令 $q(n)$ 是 A 对 MAC 预言机进行的查询次数。 为了回答 A 的第 i 个预言机查询，预言机从大小为 $2^{n/4}$ 的集合中均匀选择 r_i。 事件 $Repeat$ 的概率恰好是某个 $i \neq j$ 使得 $r_i = r_j$ 的概率。 应用**“生日界”**（**引理A.15**），**我们有** $\Pr[Repeat] \leq \binom{q(n)}{2} / 2^{n/4}$。 **由于 A 只进行多项式次查询，因此该值是**可忽略的**。

我们接下来考虑公式(4.3)右侧的最后一项。 我们认为，如果 $Mac\text{-}forge_{A,\Pi}(n) = 1$，但 $Repeat$ 没有发生，那么 $NewBlock$ **必然**发生。 也就是说， $Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge \overline{Repeat}$ 蕴含 $NewBlock$，因此

$$\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge \overline{Repeat} \wedge \overline{NewBlock}] = 0.$$

这在某种意义上是证明的**核心**。

再次令 $q = q(n)$ 表示 A 对 MAC 预言机进行的查询次数，并令 r_i 表示用于回答 A 的第 i 个预言机查询的随机标识符。 如果 $Repeat$ 没有发生，那么值 $r_1, \ldots, r_q$ 是**不同的**。 令 $\langle m, t = \langle r, t_1, \ldots \rangle \rangle$ 是 A 的输出，其中 $m = m_1, \ldots$。 如果 $r \notin \{r_1, \ldots, r_q\}$，那么 $NewBlock$ 显然发生。 如果不是，那么 $r = r_j$ 对于某个**唯一**的 j，块 $\langle r, l, 1, m_1 \rangle, \ldots$ 就**不可能**在回答任何**其他** Mac 查询（除了第 j 个查询）的过程中被认证。 令 $m^{(j)}$ 是 A 用于其第 j 个预言机查询的消息，并令 l_j 是其长度。 有两种情况需要考虑： **情况 1：** $l \neq l_j$。 在回答第 j 个 Mac 查询时被认证的块**都**具有第二个位置的 $l_j \neq l$。 因此 $\langle r, l, m_1 \rangle$ 特别是**从未**在回答第 j 个 Mac 查询的过程中被认证， $NewBlock$ 发生。

情况 2： $l = l_j$。 如果 $Mac\text{-}forge_{A,\Pi}(n) = 1$，那么我们**必须有** $m \neq m^{(j)}$。 令 $m^{(j)} = m_1^{(j)}, \ldots$。 由于 m 和 $m^{(j)}$ 长度相等，则**至少存在**一个索引 i 使得 $m_i \neq m_i^{(j)}$。 块 $\langle r, l, i, m_i \rangle$ 随后**从未**在回答第 j 个 Mac 查询的过程中被认证。 （因为 i 包含在块的第三个位置，块 $\langle r, l, i, m_i \rangle$ 只能被认证如果 $\langle r, l, i, m_i \rangle = \langle r_j, l_j, i, m_i^{(j)} \rangle$，但这不成立，因为 $m_i \neq m_i^{(j)}$。）

为了完成定理的证明，我们限制公式(4.3)右侧的第二项：

断言 4.10 $\Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge NewBlock]$ 是**可忽略的**。

该断言依赖于 Π' 的安全性。 我们构造一个攻击定长 MAC Π' 的 **ppt** 攻击者 A'，并以**非可忽略概率**成功输出了一个关于**以前未认证消息**的有效伪造。

$$\Pr[Mac\text{-}forge_{A',\Pi'}(n) = 1] \geq \Pr[Mac\text{-}forge_{A,\Pi}(n) = 1 \wedge NewBlock]. \quad (4.4)$$

Π' 的安全性意味着左侧是**可忽略的**，证明了该断言。

A' 的构造是显而易见的，因此我们简要描述它。 A' 作为子程序运行 A，并通过**自己**选择 $r \leftarrow \{0,1\}^{n/4}$，适当地解析 m，并对其自己的 MAC 预言机 $Mac'_k(\cdot)$ 进行必要的查询，来回答 A 对 m 上的标签的请求。 当 A 输出 $\langle m, t = \langle r, t_1, \ldots \rangle \rangle$ 时， A' 检查是否发生了 $NewBlock$（这很容易做到，因为 A' 可以跟踪它对自己的预言机进行的所有查询）。 如果是，则 A' 找到**第一个**块 $\langle r, l, i, m_i \rangle$，它**从未**被其自己的 MAC 预言机 Mac' 认证，并输出

$\langle\langle r, l, i, m_i\rangle, t_i\rangle$。 （如果不是， A' 不输出任何内容。）

A 作为子程序在 A' 运行时看到的视图与 A 在实验 $Mac\text{-}forge_{A,\Pi}(n)$ 中看到的视图**分布相同**，因此事件 $Mac\text{-}forge_{A,\Pi}(n) = 1$ 和 $NewBlock$ 的概率**不变**。 如果 $NewBlock$ 发生，则 A' 输出的块 $\langle r, l, i, m_i\rangle$ 以前从未被其自己的 MAC 预言机认证；如果 $Mac\text{-}forge_{A,\Pi}(n) = 1$，则每个块上的标签都是有效的（相对于 Π'），因此特别是关于 A' 输出的块上的标签。 这意味着每当 $Mac\text{-}forge_{A,\Pi}(n) = 1$ 和 $NewBlock$ 发生时，我们有 $Mac\text{-}forge_{A',\Pi'}(n) = 1$，证明了公式(4.4)。

4.4 CBC-MAC

定理4.6和4.8表明，可以从伪随机 函数构造用于任意长度消息的安全消息认证码。 这在原则上证明了安全的 MAC 可以从分组密码构造。 不幸的是，由此产生的构造**效率极低**：对于长度为 dn 的消息，分组密码被评估4 d 次；标签的长度超过 $4dn$ 位。 幸运的是，有**更有效**的构造可用。 我们在这里探讨一种依赖于**分组密码**的构造，另一种在第5.3.2节中探讨的构造使用了**额外的密码原语**。

4.4.1 基本构造

CBC-MAC 是一种**标准化**的消息认证码，在实践中被广泛使用。 CBC-MAC 的一个基本版本，当认证**任何固定长度**的消息时是安全的，如**构造 4.11** 所示。 （另见图4.1。） 我们警告，在消息长度不同时可以被认证的**一般情况下**，这个基本方案是**不安全**的；参见下面的进一步讨论。

构造 4.11 令 F 是一个伪随机函数，并固定一个长度函数 $l > 0$。 基本的 CBC-MAC 构造如下： • **Mac：** 输入密钥 $k \in \{0,1\}^n$ 和长度为 $l(n) \cdot n$ 的消息 m，执行以下操作（我们设置 $l = l(n)$）：

1. 将 m 解析为 $m = m_1, \ldots, m_l$，其中每个 m_i 的长度为 n。
2. 设置 $t_0 := 0^n$。 然后，对于 $i = 1$ 到 l： 设置 $t_i := F_k(t_{i-1} \oplus m_i)$。 输出 t_l 作为标签。 • **Vrfy：** 输入密钥 $k \in \{0,1\}^n$、消息 m 和标签 t，执行以下操作： 如果 m 的长度不是 $l(n) \cdot n$，则输出0。 否则，当且仅当 $t = Mac_k(m)$ 时输出1。 **基本 CBC-MAC**（用于**定长**消息）。

定理 4.12 令 l 是一个多项式。 如果 F 是一个伪随机函数，则**构造 4.11** 是一个用于加密长度为 $l(n) \cdot n$ 的消息的**安全 MAC**。

定理 4.12 的证明**相当复杂**。 在下一节中，我们将证明一个更一般的结果，该定理由该结果得出。 尽管**构造 4.11** 可以以显而易见的方式扩展到处理长度为 n 的任意倍数的消息，但在认证

消息的长度是**固定**且发送方和接收方**事先约定**的情况下，该构造才是安全的。 （参见练习4.13。）

与**构造 4.5** 相比，这种构造的优势在于，它也给出了一个定长 MAC，它可以认证**更长**的消息。与**构造 4.7** 相比，CBC-MAC 效率高得多，对于长度为 dn 的消息，只需要评估分组密码 d 次，并且标签的长度仅为 n。

CBC-MAC 与 CBC 模式加密的比较。 CBC-MAC 与 CBC 模式的运行非常相似。 然而，有一些重要的区别：

1. CBC 模式加密使用**随机** IV，这对安全性至关重要。 相比之下，CBC-MAC **不使用** IV（或者，可以看作是使用**固定值** $IV = 0^n$），这对安全性也至关重要。 具体来说，**使用随机 IV 的 CBC-MAC 是不安全**的。
2. 在 CBC 模式加密中，所有**中间值** t_i（在 CBC 模式加密中称为 c_i）都作为密文的一部分由加密算法输出，而在 CBC-MAC 中，**只有最终块**作为标签输出。 如果修改 CBC-MAC 以输出在计算过程中获得的所有 $\{t_i\}$，那么它就不再安全了。

在练习4.14中，要求您验证上面讨论的 CBC-MAC 的修改是**不安全**的。 这些例子说明了这样一个事实：对密码构造进行看起来**无害**的修改可能会使它们**不安全**。 人们应该始终**完全按照指定的方式**实现密码构造，并且**不**引入任何变体（除非这些变体本身已被证明是安全的）。 此外，了解正在使用的构造**至关重要**。 在许多情况下，密码库会向程序员提供一个**“CBC 函数”，但它**不区分**该函数是用于**加密**还是**消息认证**。

用于任意长度消息的安全 CBC-MAC。 我们简要描述两种修改**构造 4.11** 的方法，以**可证明安全**的方式处理**任意长度消息**。 （为简单起见，我们假设所有被认证的消息长度都是 n 的倍数，并且 $Vrfy$ 拒绝 任何长度不是 n 的倍数的消息。 在下一节中，我们处理消息可以是任意长度的更一般情况。）

1. 用其长度 $|m|$（编码为 n 比特字符串）**作为前缀**，然后对结果计算基本 CBC-MAC；参见图4.2。 这种变体的安全性来自下一节中证明的结果。 请注意，将 $|m|$ **附加**到消息末尾然后计算基本 CBC-MAC 是**不安全**的。
2. 更改方案，使得**密钥生成选择两个独立的、均匀密钥 $k_1 \in \{0,1\}^n$ 和 $k_2 \in \{0,1\}^n$**。 然后，要认证消息 m，首先使用 k_1 计算 m 的基本 CBC-MAC，并令 t 为结果；输出标签 $\hat{t} := F_{k_2}(t)$。

第二种选择的优点是不需要**事先**知道消息长度（即，在开始计算标签时）。 然而，它的缺点是使用了两个 F 的密钥。 请注意，以**两次额外应用伪随机函数**为代价，可以在计算开始时存储一个密钥 k，然后导出密钥 $k_1 := F_k(1)$ 和 $k_2 := F_k(2)$。 尽管如此，在实践中，**初始化分组**

密码密钥的操作被认为是**相对昂贵**的。 因此，要求两个不同的密钥——即使它们是**即时导出**的——也是**不那么可取**的。

图 4.1： 基本 CBC-MAC（用于定长消息）。

图 4.2： 一种用于认证任意长度消息的安全 CBC-MAC 变体。

4.4.2 *安全性证明

在本节中，我们证明 CBC-MAC 不同变体的安全性。 我们首先**总结结果**，然后给出证明的**细节**。 在开始之前，我们指出本节中的证明**相当复杂**，是为**高级读者**准备的。

在整个本节中，固定一个**密钥函数** F，对于安全参数 n，它将 n 比特密钥和 n 比特输入映射到 n 比特输出。 我们定义一个**密钥函数** CBC，对于安全参数 n，它将 n 比特密钥和 $(\{0,1\}^n)^*$ 中的输入（即长度为 n 的倍数的字符串）映射到 n 比特输出。 该函数定义为

$$CBC_k(x_1,\ldots,x_l) \stackrel{\text{def}}{=} F_k(F_k(\cdots F_k(F_k(x_1)\oplus x_2)\oplus\cdots)\oplus x_l),$$

其中 $|x_1| = \cdots = |x_l| = |k| = n$。 （我们省略了对**空字符串** CBC_k 的定义。） 请注意，CBC 正是**基本 CBC-MAC**，尽管我们在这里考虑**不同长度**的输入。

字符串集 $P \subset (\{0,1\}^n)^*$ 是**前缀自由**的，如果它不包含空字符串，并且 P 中没有字符串 X 是 P 中任何其他字符串 X' 的前缀。 我们证明：

定理 4.13 如果 F 是一个**伪随机函数**，则 CBC 是一个**伪随机函数**，只要**查询它的输入集是前缀自由的**。 形式上，对于所有查询其预言机关于**前缀自由输入集**的概率多项式时间区分器 D，存在一个**可忽略函数** $negl$ 使得

$$\left|\Pr[D^{CBC_k(\cdot)}(1^n)=1] - \Pr[D^{f(\cdot)}(1^n)=1]\right| \le negl(n),$$

其中 k 是从 $\{0,1\}^n$ 中均匀选择的， f 是从将 $(\{0,1\}^n)^*$ 映射到 $\{0,1\}^n$ 的函数集中均匀选择的。

因此，我们可以将一个用于**定长输入**的伪随机函数 F 转换为一个用于**任意长度输入**的伪随机函数 CBC（受限于可以查询的输入约束）！ 为了将此用于**消息认证**，我们采用**构造 4.5** 的想法如下：要认证消息 m，首先应用某个**编码函数** $encode$ 以获得一个（非空）字符串 $encode(m) \in (\{0,1\}^n)^*$；然后输出标签 $CBC_k(encode(m))$。 为了使其安全（参见**定理 4.6** 的证明），编码需要是**前缀自由**的，即具有以下属性：对于任何不同的（合法）消息 m_1, m_2，字符串 $encode(m_1)$ **不**是 $encode(m_2)$ 的前缀。 这意味着对于任何（合法）消息

集 $\{m_1, \ldots\}$，编码消息集 $\{encode(m_1), \ldots\}$ 是前缀自由的。

我们现在研究这个想法的两个具体应用： • **固定 l，并令合法消息集为 $\{0,1\}^{l(n)\cdot n}$。** 然后我们可以采用**平凡编码** $encode(m) = m$，它是前缀自由的，因为一个字符串不可能成为不同长度的同一字符串的前缀。 这正是**基本 CBC-MAC**，我们上面所说的意味着基本 CBC-MAC 对于任何**定长**消息都是安全的（参见**定理 4.12**）。 • **处理**任意长度（非空）消息的一种方法（技术上，长度小于 2^n 的消息）是：通过**前缀**其长度 $|m|$（编码为 n 比特字符串）来编码字符串 $m \in \{0,1\}^*$，然后**附加**所需的0，使得得到的 字符串长度是 n 的倍数。 （这基本上就是图 4.2所示的。） 这种编码是**前缀自由**的，因此我们获得了用于**任意长度消息**的安全 MAC。

本节的其余部分致力于**定理 4.13** 的证明。 在证明该定理时，我们分析 CBC 何时被**随机函数 g “加密钥”**，而不是被某个底层伪随机函数 F 的**随机密钥 k** 加密钥。 也就是说，我们考虑密钥函数 CBC_g 定义为

$$CBC_g(x_1, \ldots, x_l) \stackrel{\text{def}}{=} g(g(\cdots g(g(x_1) \oplus x_2) \oplus \cdots) \oplus x_l)$$

其中，对于安全参数 n，函数 g 将 n 比特输入映射到 n 比特输出，并且 $|x_1| = \cdots = |x_l| = n$。 请注意，这里定义的 CBC_g 并不是高效的（因为 g 的表示需要**指数**空间 n），但是，它仍然是一个定义明确的密钥函数。

我们证明，如果 g 是从 $Func_n$ 中均匀选择的，则 CBC_g 与一个将 $(\{0,1\}^n)^*$ 映射到 n 比特字符串的**随机函数**是**不可区分**的，只要查询的**输入集是前缀自由**的。 更精确地说：

断言 4.14 固定任何 $n \geq 1$。 对于所有查询其预言机关于**前缀自由**的 q 个输入集、其中最长输入包含 l 块的**区分器** D，都有

$$\left| \Pr[D^{CBC_g(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1] \right| \leq \frac{q^2 l^2}{2^n},$$

其中 g 是从 $Func_n$ 中均匀选择的， f 是从将 $(\{0,1\}^n)^*$ 映射到 $\{0,1\}^n$ 的函数集中均匀选择的。

（该断言是**无条件**的，并且不对 D 的运行时间施加任何约束。 因此，我们可以假设 D 是确定性的。） 上述蕴含了**定理 4.13**，使用了我们已经看到过的标准技术。 特别是，对于任何在多项式时间内运行的 D，我们必须有 $q(n), l(n) = poly(n)$，因此 $q(n)^2 l(n)^2 \cdot 2^{-n}$ 是**可忽略的**。

证明 (断言 4.14 的证明) 固定某个 $n \geq 1$。 证明分两步进行：我们首先定义**平滑性**的概念并证明 CBC 是平滑的；然后我们证明平滑性蕴含该断言。

令 $P = \{X_1, \ldots, X_q\}$ 是一个**前缀自由**的 q 个输入集，其中每个 X_i 都在 $(\{0,1\}^n)^*$ 中，并且 P 中最长的字符串包含 l 块（即每个 $X_i \in P$ 最多包含 l 个长度为 n 的块）。 请注意，对于任何 $t_1, \ldots, t_q \in \{0,1\}^n$ 来说，都有 $\Pr[\forall i : f(X_i) = t_i] = 2^{-nq}$ 成立，其中概率是根据 f 从将 $(\{0,1\}^n)^*$ 映射到 $\{0,1\}^n$ 的函数集中均匀选择来计算的。 我们说 CBC_g 是 (q, l, δ)**-平滑**的，如果对于如上所述的每一个前缀自由集 $P = \{X_1, \ldots, X_q\}$ 和每一个 $t_1, \ldots, t_q \in \{0,1\}^n$ 来说，都有

$$\Pr[\forall i : CBC_g(X_i) = t_i] \geq (1 - \delta) \cdot 2^{-nq},$$

其中概率是根据 $g \in Func_n$ 的均匀选择来计算的。

换句话说， CBC 是平滑的，如果对于每一个固定的输入/输出对集 $\{\langle X_i, t_i \rangle\}$，其中 $\{X_i\}$ 形成一个前缀自由集， $CBC_g(X_i) = t_i$ 对所有 i 成立的概率与 $f(X_i) = t_i$ 对所有 i 成立的概率 δ-接近（其中 g 是一个从 $\{0,1\}^n$ 到 $\{0,1\}^n$ 的随机函数， f 是一个从 $(\{0,1\}^n)^*$ 到 $\{0,1\}^n$ 的随机函数）。

断言 4.15 CBC_g 是 (q, l, δ)-平滑的，对于 $\delta = q^2 l^2 \cdot 2^{-n}$。

证明 对于任何 $X \in (\{0,1\}^n)^*$，其中 $X = x_1, \ldots$ 并且 $x_i \in \{0,1\}^n$，令 $C_g(X)$ 表示在计算 $CBC_g(X)$ 期间 g 被评估的输入集； 即，如果 $X \in (\{0,1\}^n)^m$ 则

$$C_g(X) \stackrel{\text{def}}{=} \{x_1, CBC_g(x_1) \oplus x_2, \ldots, CBC_g(x_1, \ldots, x_{m-1}) \oplus x_m\}.$$

对于 $X \in (\{0,1\}^n)^m$ 和 $X' \in (\{0,1\}^n)^{m'}$，其中 $C_g(X) = \{I_1, \ldots, I_m\}$ 和 $C_g(X') = \{I'_1, \ldots, I'_{m'}\}$，我们说 X 中存在**非平凡碰撞**如果对于某个 $i \neq j$ 有 $I_i = I_j$，并且 X 和 X' 之间存在**非平凡碰撞**如果 $I_i = I'_j$ 但 $\langle x_1, \ldots, x_i \rangle \neq \langle x'_1, \ldots, x'_j \rangle$（在后一种情况下 i 可以等于 j）。 我们说 P 中存在**非平凡碰撞**如果 P 中某个 X 中存在非平凡碰撞，或者 P 中某个 X, X' 对之间存在非平凡碰撞。 令 $Coll$ 是 P 中存在非平凡碰撞的事件。

我们分两步证明该断言。 首先，我们证明以 P 中没有非平凡碰撞为条件， $\Pr[\forall i : CBC_g(X_i) = t_i]$ 的概率**恰好**是 2^{-nq}。 接下来，我们证明 P 中存在非平凡碰撞的概率小于 $\delta = q^2 l^2 \cdot 2^{-n}$。

考虑通过**一次一个**地选择 g 在不同输入上的**均匀值**来选择均匀 g。

我们可以通过选择 g 在不同输入上的均匀值来选择均匀 g。确定两个字符串 $X, X' \in P$ 之间是否存在非平凡碰撞可以通过首先选择 $g(I_1)$ 和 $g(I'_1)$ 的值（如果 $I'_1 = I_1$，则这些值相同），然后选择 $g(I_2)$ 和 $g(I'_2)$ 的值（请注意 $I_2 = g(I_1) \oplus x_2$ 和 $I'_2 = g(I'_1) \oplus x'_2$ 在固定 $g(I_1), g(I'_1)$ 后定义），并以这种方式继续，直到我们选择 $g(I_{m-1})$ 和 $g(I'_{m-1})$ 的值。特别

是，我们不需要按顺序选择 $g(I_m), g(I'_{m'})$ 的值来确定 X 和 X' 之间是否存在非平凡碰撞。沿着这种推理思路，可以通过选择 g 在 $C_g(X_1), \ldots, C_g(X_q)$ 的所有**非最终条目**上的值来确定 $Coll$ 是否发生。

假设在固定 g 在如上所述的各种输入上的值后，$Coll$ **没有发生**。考虑 $C_g(X_1), \ldots, C_g(X_q)$ 中的最终条目。这些条目都是**不同**的（这是从 $Coll$ 没有发生这一事实立即得出的），并且我们声称 g 在这些点上的值**尚未被固定**。事实上，g 的值在这些点上可能已被固定的**唯一**方式是如果某个 $C_g(X)$ 的最终条目 I_m 等于某个 $C_g(X')$ 的**非最终条目** I'_j。但由于 $Coll$ 没有发生，这种情况**只能**在 $X' = X$ 且 $\langle x'_1, \ldots, x'_j \rangle = \langle x_1, \ldots, x_m \rangle$ 时发生。但随后 X 将是 X' 的前缀，这违反了 P 是前缀自由集这一假设。

由于 g 是一个随机函数，上述意味着 $CBC_g(X_1), \ldots, CBC_g(X_q)$ 是**均匀且相互独立**的，并且与所有其他已固定的 g 的值也是独立的。 （这是因为 $CBC_g(X_i)$ 的值是在 $C_g(X_i)$ 的最终条目上评估 g 的值，该输入值与 g 已经固定的所有其他输入是不同的。） 因此，对于任何 $t_1, \ldots, t_q \in \{0,1\}^n$，我们有

$$\Pr\left[\forall i : CBC_g(X_i) = t_i \mid Coll\right] = 2^{-nq}. \tag{4.5}$$

我们接下来证明 $Coll$ 以高概率发生，通过对 $\Pr[Coll]$ 进行**上界**限制。 对于 P 中不同的 X_i, X_j，令 $Coll_{i,j}$ 表示 X 或 X' 中存在非平凡碰撞，或者 X 和 X' 之间存在非平凡碰撞的事件。 我们有 $Coll = \bigvee_{i \neq j} Coll_{i,j}$，因此**并集界**给出

$$\Pr[Coll] \leq \sum_{i \neq j} \Pr[Coll_{i,j}] = \binom{q}{2} \cdot \Pr[Coll_{i,j}] \leq \frac{q^2}{2} \cdot \Pr[Coll_{i,j}]. \tag{4.6}$$

固定 P 中不同的 $X = X_i$ 和 $X' = X_j$，我们现在限制 $Coll_{i,j}$。 正如分析将清楚显示的那样，当 X 和 X' 都尽可能长时，概率最大化，因此我们假设它们的长度都为 l 块。 令 $X = (x_1, \ldots, x_l)$ 和 $X' = (x'_1, \ldots, x'_l)$，并令 t 是一个最大的整数，使得 $(x_1, \ldots, x_t) = (x'_1, \ldots, x'_t)$。 （注意 $t < l$，否则 $X = X'$。） 我们假设 $t > 0$，但下面的分析可以很容易地修改，对于 $t = 0$ 也能得出相同的结果。 我们继续令 $I_1, I_2, \ldots$（相应地， $I'_1, I'_2, \ldots$）表示在计算 $CBC_g(X)$（相应地， $CBC_g(X')$）期间 g 的输入；请注意 $(I'_1, \ldots, I'_t) = (I_1, \ldots, I_t)$。 考虑通过**一次一个**地选择 g 的输出的**均匀值**来选择 g。 我们按 $2l - 2$ 个步骤进行： **步骤 1 到** $t - 1$ (如果 $t > 1$): 在每个步骤 i，选择 $g(I_i)$ 的均匀值，从而定义 I_{i+1} 和 I'_{i+1}（它们是相等的）。 **步骤** t**:** 选择 $g(I_t)$ 的均匀值，从而定义 I_{t+1} 和 I'_{t+1}。 **步骤** $t+1$ **到** $l - 1$ (如果 $t < l - 1$): 依次选择 $g(I_{t+1}), g(I_{t+2}), \ldots, g(I_{l-1})$ 的均匀值，从而定义 $I_{t+2}, I_{t+3}, \ldots, I_l$。 **步骤** l **到** $2l - 2$ (如果 $t < l - 1$): 依次选择 $g(I'_{t+1}), g(I'_{t+2}), \ldots, g(I'_{l-1})$ 的均匀值，从而定义 $I'_{t+2}, I'_{t+3}, \ldots, I'_l$。

令 $Coll(k)$ 是在步骤 k 发生非平凡碰撞的事件。 然后

$$\Pr[Coll_{i,j}] = \Pr\left[\bigvee_k Coll(k)\right] \leq \Pr[Coll(1)] + \sum_{k=2}^{2l-2} \Pr[Coll(k) \mid \overline{Coll(k-1)}] \quad (4.7)$$

使用**命题 A.9**。 对于 $k < t$，我们声称 $\Pr[Coll(k) \mid \overline{Coll(k-1)}] = k/2^n$: 事实上，如果到步骤 $k-1$ 为止没有发生非平凡碰撞，则 $g(I_k)$ 的值是在步骤 k 中均匀选择的；非平凡碰撞只有在 $I_{k+1} = g(I_k) \oplus x_{k+1}$ 等于 $\{I_1, \ldots, I_k\}$ 之一时发生（它们都是不同的，因为 $Coll(k-1)$ 没有发生）。 通过类似的推理，我们有 $\Pr[Coll(t) \mid \overline{Coll(t-1)}] \leq 2t/2^n$（这里有两个值 I_{t+1}, I'_{t+1} 需要考虑；请注意它们不能彼此相等）。 最后，如前所述，对于 $k > t$，我们有 $\Pr[Coll(k) \mid \overline{Coll(k-1)}] = (k+1)/2^n$。 使用公式(4.7)，我们因此有

$$\Pr[Coll_{i,j}] \leq 2^{-n} \cdot \left(\sum_{k=2}^{t-1} k + 2t + \sum_{k=t+1}^{2l-2} (k+1) \right) = 2^{-n} \cdot \sum_{k=2}^{2l-1} k = 2^{-n} \cdot \frac{(2l+1)(2l-2)}{2} < 2l$$

从公式(4.6)我们得到 $\Pr[Coll] < q^2 \cdot l^2 \cdot 2^{-n} = \delta$。 最后，使用公式(4.5)我们看到

$$\Pr\left[\forall i : CBC_g(X_i) = t_i\right] \geq \Pr\left[\forall i : CBC_g(X_i) = t_i \mid Coll\right] \cdot \Pr[\overline{Coll}] = 2^{-nq} \cdot \Pr[\overline{Coll}] \geq (1 -$$

正如所声称的。

我们现在证明平滑性蕴含定理。 假设不失一般性， D 总是进行 q 个（不同的）查询，每个查询最多包含 l 个块。 D 可以自适应地选择其查询（即，取决于对先前查询的答案），但 D 的查询集必须是**前缀自由**的。

对于不同的 $X_1, \ldots, X_q \in (\{0,1\}^n)^*$ 和任意的 $t_1, \ldots, t_q \in \{0,1\}^n$，定义 $\alpha(X_1, \ldots, X_q; t_1, \ldots, t_q)$ 为当且仅当 D 在进行查询 $X_1, \ldots, X_q$ 并获得响应 $t_1, \ldots, t_q$ 时输出1。 （如果，例如， D 没有将 X_1 作为其第一个查询，则 $\alpha(X_1, \ldots; \ldots) = 0$。） 令 $X = (X_1, \ldots, X_q)$ 和 $t = (t_1, \ldots, t_q)$，那么我们有

$$\Pr[D^{CBC_g(\cdot)}(1^n) = 1] = \sum_{X \text{ prefix-free}; t} \alpha(X,t) \cdot \Pr[\forall i : CBC_g(X_i) = t_i]$$

$$\geq \sum_{X \text{ prefix-free}; t} \alpha(X,t) \cdot (1-\delta) \cdot \Pr[\forall i : f(X_i) = t_i]$$

$$= (1-\delta) \cdot \Pr[D^{f(\cdot)}(1^n) = 1],$$

其中 g 是从 $Func_n$ 中均匀选择的， f 是从将 $(\{0,1\}^n)^*$ 映射到 $\{0,1\}^n$ 的函数集中均匀选

择的。 这蕴含

$$\Pr[D^{f(\cdot)}(1^n)=1]-\Pr[D^{CBC_g(\cdot)}(1^n)=1]\leq \delta\cdot\Pr[D^{f(\cdot)}(1^n)=1]\leq\delta.$$

当 D 输出0时，对称的论证完成了证明。

4.5 认证加密

在第3章中，我们研究了如何在使用加密的私钥设置中获得**保密性**。 在本章中，我们展示了如何使用消息认证码来确保**完整性**。 一个人可能自然希望同时实现这两个目标，这就是我们现在转向解决的问题。

在私钥设置中，最好**始终**默认确保**保密性**和**完整性**。 事实上，在许多需要保密性的应用中，完整性也至关重要。 此外，缺乏完整性有时可能导致保密性泄露。

4.5.1 定义

像往常一样，我们首先精确定义我们希望实现的目标。 在抽象层面上，我们的目标是实现一个提供**保密性**和**完整性**的**“理想安全”**通信信道。 **追求这种定义超出了本书的范围。 相反，我们提供了一组更简单的定义，它们**分别**处理保密性和完整性。 这些定义和我们随后的分析足以理解手头的关键问题。 （然而，我们提醒读者，与加密和消息认证码相反，这个领域尚未就**认证加密**的标准术语和定义达成一致。）**

令 $\Pi_E=(Enc,Dec)$ 是一个私钥加密方案。 正如前面提到的，我们通过**分别**定义保密性和完整性来定义安全性。 我们考虑的保密性概念是我们以前见过的：我们要求 Π_E 对**选择密文攻击**是安全的，即它是 **CCA-安全的**。 （关于 CCA-安全性 的讨论和定义，请参阅第3.7节。） 我们在这里关注**选择密文攻击**是因为我们明确考虑了一个**主动攻击者**，她可以修改从一个诚实方发送到另一个诚实方的数据。 我们的完整性概念将本质上是**自适应选择消息攻击下的存在性不可伪造性**。 然而，由于 Π_E 不满足消息认证码的语法，我们引入了一个特定于这种情况的定义。 考虑为私钥加密方案 $\Pi=(Gen,Enc,Dec)$、攻击者 A 和安全参数 n 定义的以下实验：

不可伪造加密实验 $Enc\text{-}Forge_{A,\Pi}(n)$：

1. 运行 $Gen(1^n)$ 以获得密钥 k。
2. 攻击者 A 获得输入 1^n 和对**加密预言机** $Enc_k(\cdot)$ 的访问权限。 攻击者输出一个密文 c。
3. 令 $m:=Dec_k(c)$，并令 Q 表示 A 向其加密预言机询问的所有查询的集合。 当且仅当 (1) $m\neq\perp$ 并且 (2) $m\notin Q$ 时，实验的输出为1。

定义 4.16 一个私钥加密方案 Π 是**不可伪造的**，如果对于所有概率多项式时间攻击者 A 来

说，存在一个**可忽略函数** $negl$ 使得

$$\Pr[Enc\text{-}Forge_{A,\Pi}(n) = 1] \leq negl(n).$$

与我们关于**验证查询**的讨论并行（紧随定义4.2），这里一个人也可以考虑一个更强的定义，其中 A 额外被赋予了**解密预言机**的访问权限。 可以验证我们下面介绍的安全构造也满足这个更强的定义。

我们现在定义一个（安全的）**认证加密方案**。

定义 4.17 一个私钥加密方案是**认证加密方案**，如果它是 **CCA-安全** 且**不可伪造**的。

4.5.2 通用构造

认为**安全加密方案**和**安全消息认证码**的任何合理组合都应该导致**认证加密方案**是很有诱惑力的。在本节中，我们将展示情况**并非**如此。 这表明即使是**安全的密码学工具**也可以以某种导致结果不安全的方式组合，并再次强调**定义和安全性证明**的重要性。 从积极的一面来看，我们展示了如何**正确地**组合加密和消息认证以实现**联合保密性**和**完整性**。

贯穿始终，令 $\Pi_E = (Enc, Dec)$ 是一个 **CPA-安全** 加密方案，令 $\Pi_M = (Mac, Vrfy)$ 表示一个消息认证码，其中两种方案的密钥生成都只是选择一个均匀的 n 比特密钥。 有三种自然的方法来组合使用**独立密钥** k_E 和 k_M 的加密和消息认证，分别用于 Π_E 和 Π_M：

1. **加密然后认证 (Encrypt-and-authenticate):** 在这种方法中，加密和消息认证**独立地并行**计算。 也就是说，给定一个明文消息 m，发送方传输密文 $\langle c, t\rangle$，其中

   $$c \leftarrow Enc_{k_E}(m) \quad \text{and} \quad t \leftarrow Mac_{k_M}(m).$$

   接收方解密 c 以恢复 m；假设没有错误发生，然后它验证标签 t。 如果 $Vrfy_{k_M}(m, t) = 1$，则接收方输出 m；否则，它输出一个错误。
2. **认证然后加密 (Authenticate-then-encrypt):** 在这里，**首先计算** MAC 标签 t，然后将消息和标签一起加密。 也就是说，给定一个消息 m，发送方传输密文 c，计算为：

   $$t \leftarrow Mac_{k_M}(m) \quad \text{and} \quad c \leftarrow Enc_{k_E}(\langle m, t\rangle).$$

   接收方解密 c 以获得 $\langle m, t\rangle$；假设没有错误发生，然后它验证标签 t。 与之前一样，如果 $Vrfy_{k_M}(m, t) = 1$，则接收方输出 m；否则，它输出一个错误。
3. **加密然后认证 (Encrypt-then-authenticate):** 在这种情况下，**首先加密**消息，然后对结果计算 MAC 标签。 也就是说，密文是 $\langle c, t\rangle$ 对，其中：

$$c \leftarrow Enc_{k_E}(m) \quad \text{and} \quad t \leftarrow Mac_{k_M}(c).$$

（另见**构造 4.18**。） 如果 $Vrfy_{k_M}(c,t) = 1$，则接收方解密 c 并输出结果；否则，它输出一个错误。

我们分析上述每种方法，当它们用**“通用”安全组件**实例化时，即任意 CPA-安全 加密方案和任意（强）安全 MAC。 我们希望一种在**使用任何（安全）组件**时都能提供**联合保密性**和**完整性**的方法，因此我们将拒绝任何存在一个安全加密方案/MAC 的**单个反例**，导致组合不安全的方法为**“不安全”。 这种**“全有或全无”**的方法减少了实现缺陷的可能性。 具体来说，认证加密方案可能通过调用**“加密子程序”和**“消息认证子程序”**以及这些子程序（例如，当密码库更新或标准修改时，通常会发生这种情况）的实现可能会在稍后的某个时间点发生更改来实现。 实现一个其安全性取决于其组件**如何**实现（而不是它们提供的**安全性**）的方法是**危险的**。

我们强调，如果一种方法被拒绝，这并不意味着它对组件的**所有**可能实例化都是不安全的； 然而，这意味着必须在使用该方法之前**分析**并**证明**其任何实例化是安全的。

加密然后认证。 回想一下，在这种方法中，加密和消息认证是独立进行的。 给定消息 m，传输的值是 $\langle c, t \rangle$，其中

$$c \leftarrow Enc_{k_E}(m) \quad \text{and} \quad t \leftarrow Mac_{k_M}(m).$$

这种方法可能无法实现**最基本**的保密性水平。 为了看到这一点，请注意**安全 MAC 不保证任何保密性**，因此**标签 $Mac_{k_M}(m)$ 可能会泄露关于 m 的信息**给窃听者。 （作为一个平凡的例子，考虑一个安全 MAC，其中标签的第一个比特**总是**等于消息的第一个比特。） 因此，**加密然后认证**方法可能导致一个甚至**不**具有在窃听者存在下**不可区分加密性**的方案。

事实上，即使使用**标准组件**实例化，**加密然后认证**方法也可能**不安全**以抵抗**选择明文攻击**（与上一段中做作的反例不同）。 特别是，如果使用**确定性 MAC**，如 CBC-MAC，那么在消息上计算的标签（对于某个固定密钥 k_M）**每次都相同**。 这允许窃听者**识别同一消息何时被发送两次**，因此该方案**不是 CPA-安全**的。 在实践中使用的 MAC 大多是**确定性**的，因此这代表了一个**真正的问题**。

认证然后加密。 在这里，**首先计算** MAC 标签 $t \leftarrow Mac_{k_M}(m)$；然后将 $\langle m, t \rangle$ 加密，得到的 $Enc_{k_E}(\langle m, t \rangle)$ 被传输。 我们证明这种组合**也**不一定会产生一个**认证加密方案**。

实际上，我们已经遇到了一个对这种方法不安全的 **CPA-安全加密方案**：第3.7.2节中讨论的**带填充的 CBC 模式方案**。 （我们假设读者熟悉该节的内容。） 回想一下，该方案的工作原理是**首先**以特定方式填充明文（在我们的案例中将是 $\langle m, t \rangle$），使得结果是**分组长度的倍数**，然后使用 CBC 模式加密结果。 在解密过程中，如果 CBC 模式解密后检测到填充错误，则返回

“错误填充”错误。 对于**认证然后加密**，这意味着现在有两种潜在的**解密失败**来源：**填充可能不正确，或者 MAC 标签可能无法验证**。 在组合方案中，解密算法 Dec' 的工作原理示意图如下：

$$Dec'_{k_E,k_M}(c):$$

1. 计算 $\tilde{m} := Dec_{k_E}(c)$。如果检测到**填充错误**，返回“错误填充”并停止。
2. 将 $\tilde{m}$ 解析为 $\langle m, t\rangle$。如果 $Vrfy_{k_M}(m,t) = 1$，返回 m；否则，输出**“认证失败”**。

假设攻击者可以**区分**这两种错误消息，攻击者可以对上述方案应用第3.7.2节中描述的**相同选择密文攻击**来**恢复**给定密文中的整个**原始明文**。 （这是由于第3.7.2节中所示的**填充预言机攻击仅依赖于**是否能够得知是否存在**填充错误**的能力，这是该方案所泄露的。） 这种类型的攻击已在各种设置中成功地在现实世界中进行，例如在**使用认证然后加密的 IPsec 配置**中。

解决上述方案的一种方法是确保**只返回一个错误消息**，无论解密失败的来源是什么。 这是一种**不令人满意**的解决方案，原因有几个：(1) 可能存在合法的理由（例如，可用性、调试）需要**多个错误消息**；(2) 强制错误消息相同意味着该组合**不再是真正通用**的，即它要求**认证然后加密**方法的实现者了解底层 **CPA-安全** 加密方案返回的错误消息是什么；(3) 最重要的是，**确保不同的错误不能被区分**是**极其困难**的，例如，即使返回每个错误所需的时间差异也可能被攻击者用来区分它们（参见第4.2节末尾我们对**时序攻击**的讨论）。 SSL 的某些版本尝试将**认证然后加密**方法与**单一错误消息**结合使用，但使用**时序信息**的**填充预言机攻击**仍然成功进行。 我们得出结论，**认证然后加密**通常**不提供认证加密**，并且**不应该使用**。

加密然后认证。 在这种方法中，**首先加密**消息，然后对结果计算 **MAC**。 也就是说，消息是 $\langle c, t\rangle$ 对，其中

$$c \leftarrow Enc_{k_E}(m) \quad \text{and} \quad t \leftarrow Mac_{k_M}(c).$$

$\langle c, t\rangle$ 的解密是通过当且仅当 $Vrfy_{k_M}(c,t) = 1$ 时输出 $\perp$，否则输出 $Dec_{k_E}(c)$ 来完成的。参见**构造 4.18** 的正式描述。

构造 4.18 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，其中在每种情况下密钥生成都是通过简单地选择一个均匀的 n 比特密钥来完成的。定义一个私钥加密方案 $\Pi' = (Gen', Enc', Dec')$ 如下： • **Gen'：** 输入 1^n，选择独立、均匀的 $k_E, k_M \in \{0,1\}^n$ 并输出密钥 $\langle k_E, k_M\rangle$。 • **Enc'：** 输入密钥 $\langle k_E, k_M\rangle$ 和明文消息 m，计算 $c \leftarrow Enc_{k_E}(m)$ 和 $t \leftarrow Mac_{k_M}(c)$。 输出密文 $\langle c, t\rangle$。 • **Dec'：** 输入密钥 $\langle k_E, k_M\rangle$ 和密文 $\langle c, t\rangle$，首先检查 $Vrfy_{k_M}(c,t) \neq 1$。 如果是，则输出 $Dec_{k_E}(c)$；如果不

是，则输出 $\perp$。 **一个认证加密方案的通用构造。**

这种方法是**可靠的**，只要 MAC 是**强安全**的，如**定义 4.3**。 作为这种方法安全性的直觉，密文 $\langle c, t\rangle$ 是有效的，如果 t 是 c 上的有效 MAC 标签。 MAC 的**强安全性**确保攻击者将无法生成任何**有效密文**，该密文不是从其加密预言机收到的。 这立即意味着**构造 4.18** 是**不可伪造**的。 至于 **CCA-安全性**，对密文计算的 MAC 具有使**解密预言机无用**的效果，因为对于攻击者提交给其解密预言机的每个密文 $\langle c, t\rangle$，攻击者要么**已经知道**解密结果（如果它是从其自己的加密预言机收到的 $\langle c, t\rangle$），要么可以期望结果是**错误**的（因为攻击者无法生成任何新的、有效密文）。 CCA-安全性 of Π' 规约到 Π_E 的 CPA-安全性。 还要注意 MAC 在解密**之前**进行验证；因此，MAC 验证**不会泄露**关于明文的任何信息（与我们在**认证然后加密**方法中看到的**填充预言机攻击**形成对比）。 我们现在将上述论点形式化。

定理 4.19 令 Π_E 是一个 **CPA-安全私钥加密方案**，令 Π_M 是一个**强安全消息认证码**。 则**构造 4.18** 是一个**认证加密方案**。

证明 令 Π' 表示由**构造 4.18** 产生的方案。 我们需要证明 Π' 是**不可伪造**的，并且它是 **CCA-安全** 的。 遵循上面给出的直觉，密文 $\langle c, t\rangle$ 是**有效**的（相对于某个固定的秘密密钥 $\langle k_E, k_M\rangle$），如果 $Vrfy_{k_M}(c, t) = 1$。 我们证明 Π_M 的**强安全性**意味着（除了可忽略的概率）攻击者提交给解密预言机的**任何“新”密文**都是**无效**的。 正如已经讨论的那样，这立即蕴含**不可伪造性**。 （事实上，它比不可伪造性更强。） 这个事实也使得 $\Pi' = (Gen', Enc', Dec')$ 的解密预言机**无用**，并意味着 Π' 的 CCA-安全性规约到 Π_E 的 CPA-安全性。

更详细地说，令 A 是一个攻击**构造 4.18** 的**概率多项式时间攻击者**（参见定义3.33）中的**选择密文攻击**。 令密文 $\langle c, t\rangle$ 是**新**的，如果 A 没有从其**加密预言机**或作为**挑战密文**收到 $\langle c, t\rangle$。 令 $ValidQuery$ 是 A 向其解密预言机提交一个**新**密文 $\langle c, t\rangle$，该密文是**有效**的，即 $Vrfy_{k_M}(c, t) = 1$ 的事件。 我们证明：

断言 4.20 $\Pr[ValidQuery]$ 是**可忽略的**。

证明 直觉上，这是因为如果 $ValidQuery$ 发生，则攻击者在 $Mac\text{-}sforge$ 实验中**伪造**了一个新的、有效的 $\langle c, t\rangle$ 对。 形式上，令 $q(\cdot)$ 是 A 对解密预言机查询次数的**多项式上限**，并考虑以下攻击者 A_M 攻击消息认证码 Π_M（即，在实验 $Mac\text{-}sforge_{A_M, \Pi_M}(n)$ 中运行）：

攻击者 A_M： A_M 获得输入 1^n 并可以访问 MAC 预言机 $Mac_{k_M}(\cdot)$。

1. 选择均匀 $k_E \in \{0,1\}^n$ 和 $i \in \{1, \ldots, q(n)\}$。
2. 运行 A 关于输入 1^n。 当 A 进行关于消息 m 的**加密预言机查询**时，按如下方式回答： (i) 计算 $c \leftarrow Enc_{k_E}(m)$。 (ii) 查询 c 关于 MAC 预言机并接收响应 t。 将 $\langle c, t\rangle$ 返回给 A

。 挑战密文以**完全相同**的方式准备（选择均匀比特 $b \in \{0,1\}$ 来选择要加密的消息 m_b ）。 当 A 进行关于密文 $\langle c,t\rangle$ 的**解密预言机查询**时，按如下方式回答： 如果这是第 i 个解密预言机查询，输出 $\langle c,t\rangle$。 否则： (i) 如果 $\langle c,t\rangle$ 是对先前消息 m 的加密预言机查询的响应，返回 m。 (ii) 否则，返回 $\perp$。

本质上， A_M 是在“猜测” A 的第 i 个解密预言机查询将是 A 进行的**第一个新的、有效查询**。在这种情况下， A_M 输出一个关于消息 c 的有效伪造，它以前从未提交给自己的 MAC 预言机。

显然 A_M 在概率多项式时间内运行。 我们现在分析 A_M 产生一个好的伪造的概率。 关键点是，A **在** A_M **作为子程序运行时看到的视图**与**A 在实验 $PrivK^{\text{cca}}_{A,\Pi'}(n)$ 中看到的视图**在事件 $ValidQuery$ 发生**之前**是**分布相同**的。 **要看到这一点，请注意 A 的加密预言机查询（以及挑战密文的计算）是**完美模拟**的。 至于 A 的解密预言机查询，直到 $ValidQuery$ 发生之前，它们都被**正确模拟**。 在情况 (i) 中，这很明显。 至于情况 (ii)，如果提交给解密预言机的密文 $\langle c,t\rangle$ 是**新**的，那么只要 $ValidQuery$ 尚未发生，解密预言机查询的正确答案确实是 $\perp$。（请注意，情况 (i) 恰好是 $\langle c,t\rangle$ 不是新的情况，情况 (ii) 恰好是 $\langle c,t\rangle$ 是新的情况。） 回想一下 A **不允许**将挑战密文提交给解密预言机。

因为 A 在 A_M 作为子程序运行时看到的视图与 A 在实验 $PrivK^{\text{cca}}_{A,\Pi'}(n)$ 中看到的视图在事件 $ValidQuery$ 发生之前是**分布相同**的，所以事件 $ValidQuery$ 在实验 $Mac\text{-}sforge_{A_M,\Pi_M}(n)$ 中发生的概率与在实验 $PrivK^{\text{cca}}_{A,\Pi'}(n)$ 中发生的概率相同。

如果 A_M 正确猜出 $ValidQuery$ 发生时的第一个索引 i，那么 A_M 输出 $\langle c,t\rangle$，其中 $Vrfy_{k_M}(c,t)=1$（因为 $\langle c,t\rangle$ 有效）并且它从未在对 $Mac_{k_M}(c)$ 的查询中获得标签 t（因为 $\langle c,t\rangle$ 新）。 在这种情况下， A_M 在实验 $Mac\text{-}sforge_{A_M,\Pi_M}(n)$ 中成功。

A_M 正确猜出 i 的概率是 $1/q(n)$。 因此

$$\Pr[Mac\text{-}sforge_{A_M,\Pi_M}(n)=1] \geq \Pr[ValidQuery]/q(n).$$

由于 Π_M 是一个**强安全 MAC** 并且 q 是多项式，我们得出结论 $\Pr[ValidQuery]$ 是**可忽略的**。

我们使用**断言 4.20** 来证明 Π' 的安全性。 更简单的情况是证明 Π' 是**不可伪造**的。 这立即由断言得出，因此我们只提供非正式的推理，而不是正式的证明。 首先观察到，不可伪造加密实验中的攻击者 A' 是选择密文实验中攻击者的**受限版本**（在前一个实验中，攻击者只能访问加密预言机）。 当 A' 在其实验结束时输出密文 $\langle c,t\rangle$ 时，当且仅当 $\langle c,t\rangle$ **有效**且**新**时它才“成功”。 但前一个断言精确地表明这种事件的概率是**可忽略的**。

证明 Π' 是 **CCA-安全** 稍微复杂一些。 令 A 再次是一个在**选择密文攻击**中攻击 Π' 的**概率多项式时间攻击者**。 我们有

$$\Pr\left[PrivK^{\text{cca}}_{A,\Pi'}(n)=1\right] \leq \Pr[ValidQuery] + \Pr\left[PrivK^{\text{cca}}_{A,\Pi'}(n)=1 \wedge \overline{ValidQuery}\right]. \tag{4.8}$$

我们已经证明 $\Pr[ValidQuery]$ 是**可忽略的**。 因此，以下断言完成了定理的证明。

断言 4.21 存在一个**可忽略函数** $negl$ 使得

$$\Pr\left[PrivK^{\text{cca}}_{A,\Pi'}(n)=1 \wedge \overline{ValidQuery}\right] \leq \frac{1}{2} + negl(n).$$

为了证明这个断言，我们依赖于 Π_E 的 CPA-安全性。 考虑以下攻击者 A_E 攻击 Π_E 的**选择明文攻击**：

攻击者 A_E： A_E 获得输入 1^n 并可以访问 $Enc_{k_E}(\cdot)$。

1. 选择均匀 $k_M \in \{0,1\}^n$。
2. 运行 A 关于输入 1^n。 当 A 进行关于消息 m 的**加密预言机查询**时，按如下方式回答： (i) 查询 m 关于 $Enc_{k_E}(\cdot)$ 并接收响应 c。 (ii) 计算 $t \leftarrow Mac_{k_M}(c)$ 并将 $\langle c,t\rangle$ 返回给 A。 当 A 进行关于密文 $\langle c,t\rangle$ 的**解密预言机查询**时，按如下方式回答： • 如果 $\langle c,t\rangle$ 是对先前消息 m 的**加密预言机查询**的响应，返回 m。 • 否则，返回 $\perp$。
3. 当 A 输出消息 $\langle m_0,m_1\rangle$ 时，输出**相同的消息**，并接收挑战密文 c 作为响应。 计算 $t \leftarrow Mac_{k_M}(c)$，并将 $\langle c,t\rangle$ 作为挑战密文返回给 A。 继续像上面一样回答 A 的预言机查询。
4. 输出 A 输出的比特 b'。

请注意， A_E 不需要解密预言机，因为它简单地假设 A 的任何解密查询，如果不是先前**加密预言机查询**的结果，则为**无效**。

显然 A_E 在概率多项式时间内运行。 此外，A **在** A_E **作为子程序运行时看到的视图**与**A 在实验 $PrivK^{\text{cca}}_{A,\Pi'}(n)$ 中看到的视图**在事件 $ValidQuery$ 不发生时是**分布相同**的**。 因此， A_E 成功时 $ValidQuery$ 不发生的概率与 A 成功时 $ValidQuery$ 不发生的概率相同；即，

$$\Pr\left[PrivK^{\text{cpa}}_{A_E,\Pi_E}(n)=1 \wedge \overline{ValidQuery}\right] = \Pr\left[PrivK^{\text{cca}}_{A,\Pi'}(n)=1 \wedge \overline{ValidQuery}\right],$$

这蕴含着

$$\Pr\left[PrivK^{\text{cpa}}_{A_E,\Pi_E}(n)=1\right] \geq \Pr\left[PrivK^{\text{cpa}}_{A_E,\Pi_E}(n)=1 \wedge \overline{ValidQuery}\right] = \Pr\left[PrivK^{\text{cca}}_{A,\Pi'}(n)=

由于 Π_E 是 CPA-安全的，存在一个**可忽略函数** $negl$ 使得 $\Pr[PrivK^{\text{cpa}}_{A_E,\Pi_E}(n)=1] \leq 1/2+negl(n)$。 这证明了该断言。

对独立密钥的需求。 我们以强调密码学的一个基本原则来结束本节：**不同的密码学原语实例应始终使用独立密钥**。 为了在这里说明这一点，考虑当**相同的密钥** k 用于加密和认证时，**加密然后认证**方法可能发生的情况。 令 F 是一个**强伪随机置换**。 由此可知 F^{-1} 也是一个强伪随机置换。 定义 $Enc_k(m)=F_k(\langle m,r\rangle)$ 对于 $m\in\{0,1\}^{n/2}$ 和均匀 $r\in\{0,1\}^{n/2}$，并定义 $Mac_k(c)=F_k^{-1}(c)$。 可以证明这种加密方案是 CPA-安全的（事实上，它甚至是 CCA-安全的；参见练习4.25），并且我们知道给定的消息认证码是安全 MAC。 然而，使用相同密钥 k 应用于消息 m 的**加密然后认证**组合产生：

$$Enc_k(m),Mac_k(Enc_k(m))=F_k(\langle m,r\rangle),F_k^{-1}(F_k(\langle m,r\rangle))=\langle F_k(\langle m,r\rangle),\langle m,r\rangle\rangle,$$

并且**消息 m 在明文中被泄露**！ 这并没有以任何方式与**定理 4.19** 相矛盾，因为**构造 4.18** 明确要求 k_M,k_E 是（均匀地和）**独立地**选择的。 我们鼓励读者检查在**定理 4.19** 的证明中，**这种独立性在哪里被使用了**。

4.6 *信息理论 MAC

在前面的章节中，我们探讨了具有**计算安全性**的消息认证码，即其中对攻击者的运行时间进行了限制。 回顾第2章的结果，很自然地会问，在**无界攻击者**存在下的消息认证是否是**信息理论上的**可能性。

在本节中，我们展示了在什么条件下可以实现**信息理论上的**（与计算上的相对）安全性。 第一个观察是，在这种情况下**不可能**实现**“完美”安全性**：也就是说，我们不能指望消息认证码的概率是0，即攻击者对以前未认证的消息输出一个有效标签。 原因是攻击者可以简单地**猜测**任何消息上的有效标签 t，并且猜测正确的概率（至少）是 $1/2^{|t|}$，其中 $|t|$ 表示方案的标签长度。

上面的例子告诉我们**可以希望实现什么**：一个标签长度为 $|t|$ 的 MAC，其中伪造的概率**至多**为 $1/2^{|t|}$，即使对于**无界攻击者**也是如此。 我们将看到这是可以实现的，但仅在对**诚实方认证的消息数量**进行限制的情况下。

我们首先定义消息认证码的**信息理论安全性**。 一个起点是采用用于定义**计算安全 MAC** 的实验 $Mac\text{-}forge_{A,\Pi}(n)$（参见**定义 4.2**），但是**删除安全参数** n，并要求对于**所有**（甚至**无界**）攻击者 A 来说， $\Pr[Mac\text{-}forge_{A,\Pi}=1]$ 应该**“小”。 正如前面提到的（并且将在第4.6.2节中正式证明），这样的定义是**不可能**实现的。 相反，信息理论上的安全性只有在我们对**诚实方认证的消息数量**进行一些限制时才能实现。 我们在这里看**最基本**的设置，即双方只认

证单个消息**。 我们将此称为**一次性消息认证**。 以下实验修改了 $Mac\text{-}forge_{A,\Pi}(n)$，遵循上述讨论：

一次性消息认证实验 $Mac\text{-}forge^{\text{1-time}}_{A,\Pi}$：

1. 通过运行 Gen 生成一个密钥 k。
2. 攻击者 A 输出一个消息 m'，并获得一个标签 $t' \leftarrow Mac_k(m')$ 作为响应。
3. A 输出 $\langle m, t \rangle$。
4. 当且仅当 (1) $Vrfy_k(m,t) = 1$ 并且 (2) $m \neq m'$ 时，实验的输出定义为1。

定义 4.22 一个消息认证码 $\Pi = (Gen, Mac, Vrfy)$ 是**一次性 ε-安全的**，或简称为 **ε-安全**的，如果对于所有（甚至**无界**）攻击者 A 来说：

$$\Pr[Mac\text{-}forge^{\text{1-time}}_{A,\Pi} = 1] \leq \varepsilon.$$

4.6.1 构建信息理论 MAC

在本节中，我们展示了如何基于任何**强通用函数**构建一个 ε-安全 MAC。 然后，我们展示了后者的一个简单构造。

令 $h : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ 是一个密钥函数，其第一个输入是密钥 $k \in \mathcal{K}$，第二个输入取自某个域 $\mathcal{M}$。 像往常一样，我们写 $h_k(m)$ 代替 $h(k,m)$。 当 k 是一个**均匀密钥**时，如果对于任何两个**不同**的输入 m, m'，值 $h_k(m)$ 和 $h_k(m')$ 在 $\mathcal{T}$ 中是**均匀且独立**分布的，则 h 是**强通用**的（或**成对独立**的）。 这等价于说， $\Pr[h_k(m) = t \wedge h_k(m') = t']$ 对于任何特定的值 t, t' **恰好**是 $1/|\mathcal{T}|^2$。 也就是说：

定义 4.23 一个函数 $h : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ 是**强通用**的，如果对于所有**不同**的 $m, m' \in \mathcal{M}$ 和所有 $t, t' \in \mathcal{T}$ 来说，都有

$$\Pr[h_k(m) = t \wedge h_k(m') = t'] = \frac{1}{|\mathcal{T}|^2},$$

其中概率是根据 $k \in \mathcal{K}$ 的均匀选择来计算的。

上述内容应该激励我们从任何**强通用函数** h 构造一个**一次性消息认证码**。 消息 m 上的标签 t 是通过计算 $h_k(m)$ 获得的，其中密钥 k 是均匀的；参见**构造 4.24**。 直觉上，即使攻击者观察到任何消息 m' 的标签 $t' := h_k(m')$，从攻击者的角度来看，任何其他消息 m 的正确标签 $h_k(m)$ 在 $\mathcal{T}$ 中仍然是**均匀分布**的。 因此，攻击者只能盲目猜测标签，并且只有以 $1/|\mathcal{T}|$ 的概率才能猜对。

构造 4.24 令 $h : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ 是一个**强通用函数**。 定义一个用于消息 $\mathcal{M}$ 的 MAC 如下：• **Gen：** 选择均匀 $k \in \mathcal{K}$ 并输出它作为密钥。 • **Mac：** 输入密钥 $k \in \mathcal{K}$ 和消息 $m \in \mathcal{M}$，输出标签 $t := h_k(m)$。 • **Vrfy：** 输入密钥 $k \in \mathcal{K}$、消息 $m \in \mathcal{M}$ 和标签 $t \in \mathcal{T}$，当且仅当 $t = h_k(m)$ 时输出1。 （如果 $m \notin \mathcal{M}$，则输出0。） **一个从任何强通用函数构造的 MAC。**

上述构造可以被视为类似于**构造 4.5**。 这是因为**强通用函数** h 与**随机函数**是**相同**的，只要它只被评估**两次**。

定理 4.25 令 $h : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ 是一个**强通用函数**。 则**构造 4.24** 是一个关于消息 $\mathcal{M}$ 的 $1/|\mathcal{T}|$**-安全 MAC**。

证明 令 A 是一个攻击者。 在**信息理论设置**中，像往常一样，我们可以假设 A 是**确定性**的，不失一般性。 因此 A 在实验开始时输出的消息 m' 是固定的。 此外， A 在实验结束时输出的对 $\langle m, t\rangle$ 是 m' 上的标签 t' 的**确定性函数**， A 接收到。 因此我们有

$$\begin{aligned}\Pr[Mac\text{-}forge^{\text{1-time}}_{A,\Pi} = 1] &= \sum_{t' \in \mathcal{T}} \Pr\left[Mac\text{-}forge^{\text{1-time}}_{A,\Pi} = 1 \wedge h_k(m') = t'\right] \\ &= \sum_{t' \in \mathcal{T} \text{ s.t. } \langle m,t\rangle := A(t')} \Pr[h_k(m) = t \wedge h_k(m') = t'] \\ &= \sum_{t' \in \mathcal{T} \text{ s.t. } \langle m,t\rangle := A(t')} \frac{1}{|\mathcal{T}|^2} = \frac{1}{|\mathcal{T}|}.\end{aligned}$$

这证明了定理。

我们现在转向一个**强通用函数**的**经典构造**。 我们假设读者对模素数运算的一些基本知识；读者可以参考第8.1.1节和8.1.2节以获取必要的背景知识。 固定某个**素数** p，并令 $\mathbb{Z}_p \stackrel{\text{def}}{=} \{0, \ldots, p-1\}$。 我们以 $\mathcal{M} = \mathbb{Z}_p$ 作为我们的**消息空间**；可能的**标签空间**也将是 $\mathcal{T} = \mathbb{Z}_p$。 一个密钥 $\langle a, b\rangle$ 由 $\mathbb{Z}_p$ 中的一对元素组成；因此 $\mathcal{K} = \mathbb{Z}_p \times \mathbb{Z}_p$。 定义 h 为

$$h_{a,b}(m) \stackrel{\text{def}}{=} [a \cdot m + b \bmod p],$$

其中符号 $[X \bmod p]$ 指的是整数 X 模 p 的余数（因此 $[X \bmod p] \in \mathbb{Z}_p$ 始终成立）。

定理 4.26 对于任何素数 p，函数 h 是**强通用**的。

证明 固定任何**不同**的 $m, m' \in \mathbb{Z}_p$ 和任何 $t, t' \in \mathbb{Z}_p$。 对于哪些密钥 $\langle a, b\rangle$ 满足 $h_{a,b}(m) =$

t 且 $h_{a,b}(m') = t'$？ 这仅在以下情况下成立

$$a \cdot m + b = t \bmod p \quad \text{and} \quad a \cdot m' + b = t' \bmod p.$$

因此，我们在两个未知数 a, b 中有两个**线性方程**。 当且仅当 $a = [(t - t') \cdot (m - m')^{-1} \bmod p]$ 且 $b = [t - a \cdot m \bmod p]$ 时，这两个方程都被**精确满足**；请注意 $[(m - m')^{-1} \bmod p]$ 存在，因为 $m \neq m'$，因此 $m - m' \neq 0 \bmod p$。 重述一下，这意味着对于任何如上所述的 m, m', t, t'，都存在**唯一**密钥 $\langle a, b \rangle$ 使得 $h_{a,b}(m) = t$ 且 $h_{a,b}(m') = t'$。 由于有 $|\mathcal{T}|^2$ 个密钥，我们得出结论，概率（在密钥的选择上） $h_{a,b}(m) = t$ 且 $h_{a,b}(m') = t'$ 恰好是 $1/|\mathcal{K}| = 1/|\mathcal{T}|^2$，正如所要求的那样。

构造 4.24 的参数。 我们简要讨论当用上述**强通用函数**实例化**构造 4.24** 时的参数，忽略 p 不是2的幂这一事实。 该构造是一个 $1/|\mathcal{T}|$**-安全 MAC**，标签长度为 $\log |\mathcal{T}|$；标签长度对于所实现的安全级别是**最优的**。

令 $\mathcal{M}$ 是我们想要构造**一次性安全 MAC** 的某个**固定消息空间**。 上面的构造提供了一个 $1/|\mathcal{M}|$-安全 MAC，其密钥的长度是消息长度的**两倍**。 读者可能会注意到这里有两个问题，处于光谱的两端： 首先，如果 $|\mathcal{M}|$ 很**小**，那么 $1/|\mathcal{M}|$ 的伪造概率可能**不可接受**地大。 另一方面，如果 $|\mathcal{M}|$ 很**大**，那么 $1/|\mathcal{M}|$ 的伪造概率可能**过度杀伤**；一个人可能愿意接受（稍微）更大的伪造概率，如果该安全级别可以用**更短的密钥**实现。 第一个问题（当 $|\mathcal{M}|$ 小时）很容易处理，只需将 $\mathcal{M}$ 嵌入到一个更大的消息空间 $\mathcal{M}'$ 中，例如通过用0填充消息。 第二个问题也可以解决；参见本章末尾的参考文献。

4.6.2 信息理论 MAC 的限制

在本节中，我们探讨**信息理论消息认证**的限制。 我们证明任何 2^{-n}**-安全 MAC** 的密钥长度**必须至少为** $2n$。 该证明的一个扩展表明，任何 l**-次** 2^{-n}**-安全 MAC**（其中安全性是通过**定义 4.23** 的自然修改来定义的）**需要**密钥长度**至少为** $(l + 1) \cdot n$。 一个推论是**没有具有有界长度密钥的 MAC** 可以在认证**无界数量的消息**时提供**信息理论安全性**。

在下文中，我们假设消息空间包含**至少两个消息**；如果不是，则通信甚至认证都没有意义。

定理 4.27 令 $\Pi = (Gen, Mac, Vrfy)$ 是一个 2^{-n}**-安全 MAC**，其中 Gen 输出的所有密钥长度**相同**。 则 Gen 输出的密钥长度**必须至少为** $2n$。

证明 固定消息空间中的两个**不同**消息 m_0, m_1。 该证明的直觉是， m_0 上的标签必须**至少有** 2^n **种可能性**（否则攻击者可以以大于 2^{-n} 的概率猜测它）； 此外，即使以 m_0 的标签值为条件， m_1 的标签也必须有** 2^n 种可能性**（否则攻击者可以以大于 2^{-n} 的概率伪造 m_1 上的

标签）。 由于每个密钥定义了 m_0 和 m_1 上的标签，这意味着**必须至少有 $2^n \times 2^n$ 个密钥**。 我们将在下面正式说明这一点。

令 $\mathcal{K}$ 表示**密钥空间**（即 Gen 可以输出的所有可能密钥的集合）。 对于任何可能的标签 t_0，令 $\mathcal{K}(t_0)$ 表示密钥集，对于该密钥集 t_0 是 m_0 上的有效标签；即

$$\mathcal{K}(t_0) \stackrel{\text{def}}{=} \{k \mid Vrfy_k(m_0, t_0) = 1\}.$$

对于任何 t_0，我们必须有 $|\mathcal{K}(t_0)| \leq 2^{-n} \cdot |\mathcal{K}|$。 否则，攻击者可以简单地输出 $\langle m_0, t_0 \rangle$ 作为她的伪造；这将是以至少 $|\mathcal{K}(t_0)|/|\mathcal{K}| > 2^{-n}$ 的概率发生的有效伪造，这与声称的安全性相矛盾。

现在考虑攻击者 A，她请求消息 m_0 上的标签，收到标签 t_0 作为返回，选择一个均匀密钥 $k \in \mathcal{K}(t_0)$，并输出 $\langle m_1, Mac_k(m_1) \rangle$ 作为她的伪造。 A 输出有效伪造的概率**至少**是

$$\sum_{t_0} \Pr[Mac_k(m_0) = t_0] \cdot \frac{1}{|\mathcal{K}(m_0, t_0)|} \geq \sum_{t_0} \Pr[Mac_k(m_0) = t_0] \cdot \frac{2^n}{|\mathcal{K}|} = \frac{2^n}{|\mathcal{K}|}.$$

根据该方案声称的安全性，攻击者可以输出有效伪造的概率**至多**为 2^{-n}。 因此，我们必须有 $|\mathcal{K}| \geq 2^{2n}$。 由于所有密钥具有相同的长度，每个密钥的长度**必须至少为 $2n$**。

参考文献和附加阅读

消息认证码的安全定义由 Bellare 等人 从 Goldwasser 等人 给出的数字签名安全定义（参见第12章）改编而来。 关于允许验证查询的定义变体的更多信息，请参见。

使用伪随机函数进行消息认证的范式（如**构造 4.5**）由 Goldreich 等人 引入。 **构造 4.7** 归功于 Goldreich。

CBC-MAC 在1980年代早期被标准化，并且至今仍被广泛使用。 基本的 CBC-MAC（用于认证定长消息）被 Bellare 等人 证明是安全的。 Bernstein 给出了一种更直接（尽管可能不太直观）的证明，并讨论了 CBC-MAC 的一些广义版本。 正如本章所指出的，当用于认证不同长度的消息时，基本的 CBC-MAC 是不安全的。 解决这个问题的一种方法是**在消息前缀其长度**。 这具有无法应对**流数据**的缺点，即消息的长度事先未知。 Petrank 和 Rackoff 提出了一种替代的 **“在线”**方法来解决这个问题。 Black 和 Rogaway 以及 Iwata 和 Kurosawa 提供了进一步的改进；这些导致了一个新的提议标准，称为 CMAC。

认证加密的重要性首次在 中明确强调，他们提出了与我们在此处给出的定义类似的定义。 Bellare 和 Namprempre 分析了我们在此讨论的三种通用方法，尽管使用**加密然后认证**来实

现 CCA-安全性至少可以追溯到 Dolev 等人 的工作。 Krawczyk 研究了实现保密性和认证的其他方法，并分析了 SSL 使用的**认证然后加密**方法。 Degabriele 和 Paterson 展示了 IPsec 在配置为**认证然后加密**时受到攻击（认证加密的默认设置实际上是**加密然后认证**；然而，在某些配置中可以实现**认证然后加密**）。 还提出了几种**非通用**的认证加密方案；参见 以获得详细比较。

信息理论 MAC 最初由 Gilbert 等人 研究。 Wegman 和 Carter 引入了**强通用函数**的概念，并指出它们可应用于**一次性消息认证**。 他们还展示了如何通过使用**几乎强通用函数**来减少这项任务的密钥长度。 具体来说，我们在此给出的构造实现了**2^{-n}-安全性**，用于长度为 n 的消息，密钥长度为 $O(n)$； Wegman 和 Carter 展示了如何构造用于长度为 l 的消息的 **2^{-n}-安全 MAC**，密钥长度为（基本上） $O(n \cdot \log l)$。 我们在此给出的**强通用函数**的简单构造（有微小差异）归功于 Carter 和 Wegman。 有兴趣了解更多关于信息理论 MAC 的读者可参考 Stinson 的论文、Simmons 的调查，或 Stinson 教科书的第一版 [167, Chapter 10]。
