数字签名 $K_B^-(m)$ 是否满足了可鉴别的、不可伪造的需求？假设 Alice 有 $m$ 和 $K_B^-(m)$。她要在法庭上证明（进行诉讼）Bob 确实签署过这个文档，他就是唯一能够签署该文档的人。Alice 持有 Bob 的公钥 $K_B^+$，并把它用于 Bob 的数字签名 $K_B^-(m)$ 上，从而得到了文档 $m$。也就是说，Alice 计算 $K_B^+(K_B^-(m))$，瞧！在 Alice 经历了令人注目的慌乱后得到了 $m$，它与初始文档完全一致。然后，Alice 就可以论证仅有 Bob 能够签署这个文档，基于如下理由：

- 无论是谁签署这个报文，都必定在计算签名 $K_B^-(m)$ 过程中使用了 $K_B^-$ 这个私钥，使 $K_B^+(K_B^-(m))=m$。
- 知道 $K_B^-$ 这个私钥的唯一人只有 Bob。从 8.2 节我们对 RSA 的讨论中可知，知道公钥 $K_B^+$ 无助于得知私钥 $K_B^-$ 的信息。因此，知道私钥 $K_B^-$ 的人才是生成密钥对（$K_B^+$，$K_B^-$）的人，而这个人首当其冲就是 Bob。（注意到此处假设 Bob 没有把 $K_B^-$ 泄露给任何人，也没有人从 Bob 处窃取到 $K_B^-$。）

注意到下列问题是重要的，如果源文档 $m$ 被修改过，比如改成了另一个文档 $m'$，则 Bob 对 $m$ 生成的签名对 $m'$ 无效，因为 $K_B^+(K_B^-(m))$ 不等于 $m'$。因此我们看到数字签名也提供完整性，使得接收方验证该报文未被篡改，同时也验证了该报文的源。

对用加密进行数据签名的担心是，加密和解密的计算代价昂贵。给定加解密的开销，通过完全加密/解密对数据签名是杀鸡用牛刀。更有效的方法是将散列函数引入数字签名。8.3.2 节中讲过，一种散列算法取一个任意长的报文 $m$，计算生成该报文的一个固定长度的数据“指纹”，表示为 $H(m)$。使用散列函数，Bob 对报文的散列签名而不是对报文本身签名，即 Bob 计算 $K_B^-(H(m))$。因为 $H(m)$ 通常比报文 $m$ 小得多，所以生成数字签名所需要的计算耗费大为降低。

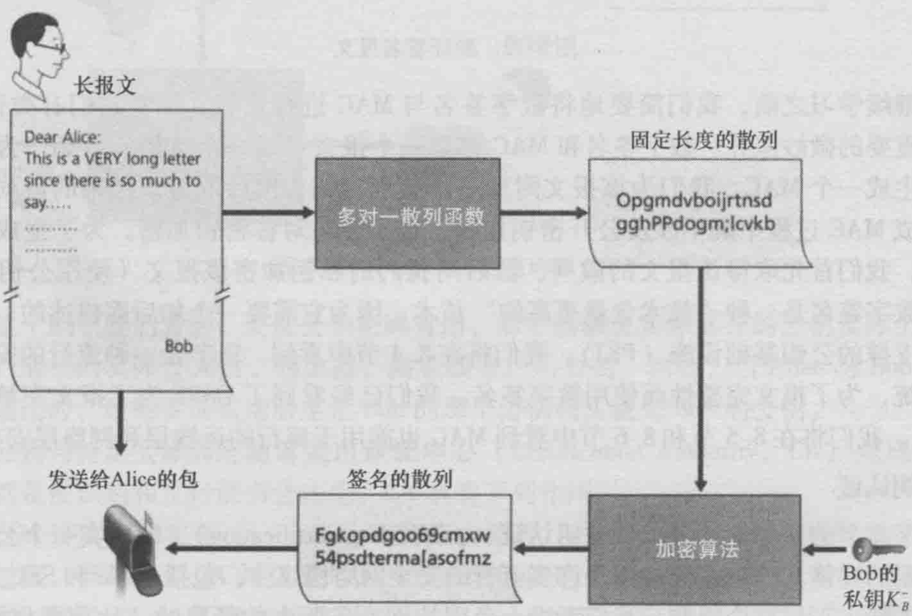


图 8-11　发送数字签名的报文

在 Bob 向 Alice 发送一个报文的情况下，图 8-11 提供了生成一个数字签名的操作过程的概览。Bob 让他的初始长报文通过一个散列函数。然后他用自己的私钥对得到的散列进行数字签名。明文形式的初始报文连同已经数字签名的报文摘要（从此以后可称为数字签