每个块由 8 比特组成。每个 8 比特块由一个“8 比特到 8 比特”表处理，这是个可管理的长度。例如，第一个块由标志为 T_1 的表来处理。接下来，这 8 个输出块被重新装配成一个 64 比特的块。该输出被回馈到 64 比特的输入，开始了第二次循环。经 n 次这样的循环后，该函数提供了一个 64 比特的密文块。这种循环的目的是使得每个输入比特影响最后输出比特的大部分（即使不是全部）。（如果仅使用一次循环，一个给定的输入比特将仅影响 64 输出比特中的 8 比特。）这种块密码算法的密钥将是 8 张排列表（假定置乱函数是公共已知的）。

目前有一些流行的块密码，包括 DES（Data Encryption Standard，数据加密标准）、3DES 和 AES（Advanced Encryption Standard，高级加密标准）。这些标准中的每种都使用了函数（而不是预先决定的表），连同图 8-5 的线（虽然对每种密码来说更为复杂和具体）。这些算法中的每种也使用了比特串作为密钥。例如，DES 使用了具有 56 比特密钥的 64 比特块。AES 使用 128 比特块，能够使用 128、192 和 256 比特长的密钥进行操作。一个算法的密钥决定了特定“小型表”映射和该算法内部的排列。对这些密码中的每种进行蛮力攻击要循环通过所有密钥，用每个密钥应用解密算法。观察到采用长度为 n 的密钥，有 2^n 种可能的密钥。NIST［NIST 2001］估计，如果用 1 秒钟破解 56 比特 DES 的计算机（就是说，每秒钟尝试所有 2^{56} 个密钥）来破解一个 128 比特的 AES 密钥，要用大约 149 万亿年的时间才有可能成功。

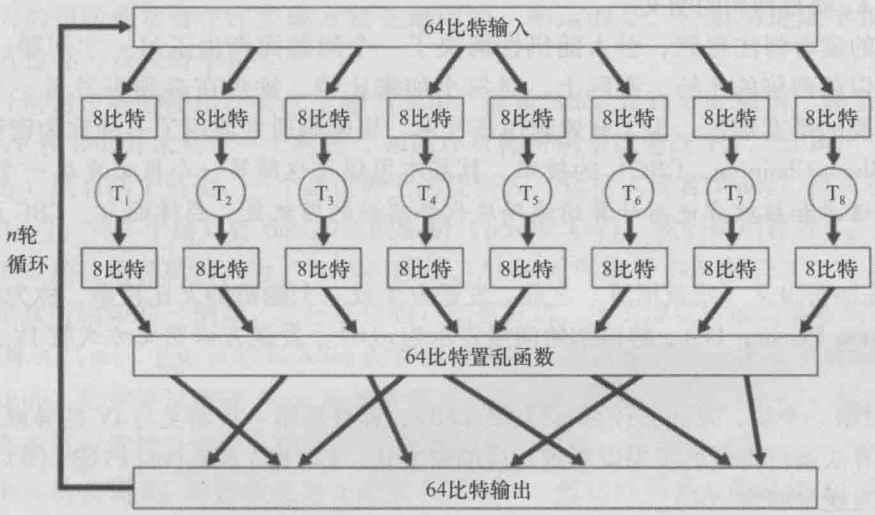

图 8-5　一个块密码的例子

2. 密码块链接

在计算机网络应用中，我们通常需要加密长报文（或长数据流）。如果我们使用前面描述的块密码，通过直接将报文切割成 k 比特块并独立地加密每块，将出现一个微妙而重要的问题。为了理解这个问题，注意到两个或更多个明文块可能是相同的。例如，两个或更多块中的明文可能是“HTTP/1.1”。对于这些相同的块，块密码当然将产生相同的密文。当攻击者看到相同的密文块时，它可能潜在地猜出其明文，并且通过识别相同的密文块和利用支撑协议结构的知识，甚至能够解密整个报文［Kaufman 1995］。

为了解决这个问题，我们能够在密文中混合某些随机性，使得相同的明文块产生不同