forged. Rather, at the beginning (or end) of the message Alice transmits $f_B f_A^{-1}(P)$. Then, when Bob deciphers the whole message, including this part, by applying f_B^{-1}, he finds that everything has become plaintext except for a small section of jibberish, which is $f_A^{-1}(P)$. Since Bob knows that the message is claimed to be from Alice, he applies f_A (which he knows, since Alice's enciphering key is public), and obtains P. Since no one other than Alice could have applied the function f_A^{-1} which is inverted by f_A, he knows that the message was from Alice.

Hash functions. A common way to sign a document is with the help of a *hash function*. Roughly speaking, a hash function is an easily computable map $f : x \mapsto h$ from a very long input x to a much shorter output h (for example, from strings of about 10^6 bits to strings of 150 or 200 bits) that has the following property: *it is not computationally feasible to find two different inputs x and x' such that $f(x') = f(x)$*. If part of Alice's "signature" consists of the hash value $h = f(x)$, where x is the entire text of her message, then Bob can verify not only that the message was really sent by Alice, but also that it wasn't tampered with during transmission. Namely, Bob applies the hash function f to his deciphered plaintext from Alice, and checks that the result agrees with the value h in Alice's signature. By assumption, no tamperer would have been able to change x without changing the value $h = f(x)$.

Key exchange. In practice, the public key cryptosystems for sending messages tend to be slower to implement than the classical systems that are in current use. The number of plaintext message units per second that can be transmitted is less. However, even if a network of users feels attached to the traditional type of cryptosystem, they may want to use a public key cryptosystem in an auxiliary capacity to send one another their keys $K = (K_E, K_D)$ for the classical system. Thus, the ground rules for the classical cryptosystem can be agreed upon, and keys can be periodically exchanged, using the slower public key cryptography; while the large volume of messages would then be sent by the faster, older methods.

Probabilistic Encryption. Most of the number theory based cryptosystems for message transmission are *deterministic*, in the sense that a given plaintext will always be encrypted into the same ciphertext any time it is sent. However, deterministic encryption has two disadvantages: (1) if an eavesdropper knows that the plaintext message belongs to a small set (for example, the message is either "yes" or "no"), then she can simply encrypt all possibilities in order to determine which is the supposedly secret message; and (2) it seems to be very difficult to *prove* anything about the security of a system if the encryption is deterministic. For these reasons, *probabilistic encryption* was introduced. We will not discuss this further or give examples in this book. For more information, see the fundamental papers on the subject by Goldwasser and Micali (*Proc. 14th ACM Symp. Theory of Computing*, 1982, 365–377, and *J. Comput. System Sci.* **28** (1984), 270–299).