Hash Function

A variation on the message authentication code is the one-way hash function. As with the message authentication code, a hash function accepts a variable-size message M as input and produces a fixed-size output, referred to as a hash code $H(M)$. Unlike a

Table 11.2 Basic Uses of Message Authentication Code C (see Figure 11.4)

$A \rightarrow B: M \| C_K(M)$ • Provides authentication —Only A and B share K (a) Message authentication
$A \rightarrow B: E_{K_2}[M \| C_{K_1}(M)]$ • Provides authentication —Only A and B share K_1 • Provides confidentiality —Only A and B share K_2 (b) Message authentication and confidentiality: authentication tied to plaintext
$A \rightarrow B: E_{K_2}[M] \| C_{K_1}(E_{K_2}[M])$ • Provides authentication —Using K_1 • Provides confidentiality —Using K_2 (c) Message authentication and confidentiality: authentication tied to ciphertext

MAC, a hash code does not use a key but is a function only of the input message. The hash code is also referred to as a message digest or hash value. The hash code is a function of all the bits of the message and provides an error-detection capability: A change to any bit or bits in the message results in a change to the hash code.

Figure 11.5 illustrates a variety of ways in which a hash code can be used to provide message authentication, as follows:

a. The message plus concatenated hash code is encrypted using symmetric encryption. This is identical in structure to the internal error control strategy shown in Figure 11.2a. The same line of reasoning applies: Because only A and B share the secret key, the message must have come from A and has not been altered. The hash code provides the structure or redundancy required to achieve authentication. Because encryption is applied to the entire message plus hash code, confidentiality is also provided.

b. Only the hash code is encrypted, using symmetric encryption. This reduces the processing burden for those applications that do not require confidentiality. Note that the combination of hashing and encryption results in an overall function that is, in fact, a MAC (Figure 11.4a). That is, $E_K[H(M)]$ is a function of a variable-length message M and a secret key K, and it produces a fixed-size output that is secure against an opponent who does not know the secret key.

c. Only the hash code is encrypted, using public-key encryption and using the sender's private key. As with (b), this provides authentication. It also provides a digital signature, because only the sender could have produced the encrypted hash code. In fact, this is the essence of the digital signature technique.

d. If confidentiality as well as a digital signature is desired, then the message plus the public-key-encrypted hash code can be encrypted using a symmetric secret key. This is a common technique.

e. This technique uses a hash function but no encryption for message authentication. The technique assumes that the two communicating parties share a common secret value S. A computes the hash value over the concatenation of M and S and appends the resulting hash value to M. Because B possesses S, it can recompute the hash value to verify. Because the secret value itself is not sent, an opponent cannot modify an intercepted message and cannot generate a false message.

f. Confidentiality can be added to the approach of (e) by encrypting the entire message plus the hash code.

When confidentiality is not required, methods (b) and (c) have an advantage over those that encrypt the entire message in that less computation is required. Nevertheless, there has been growing interest in techniques that avoid encryption (Figure 11.5e). Several reasons for this interest are pointed out in [TSUD92]:

- Encryption software is quite slow. Even though the amount of data to be encrypted per message is small, there may be a steady stream of messages into and out of a system.

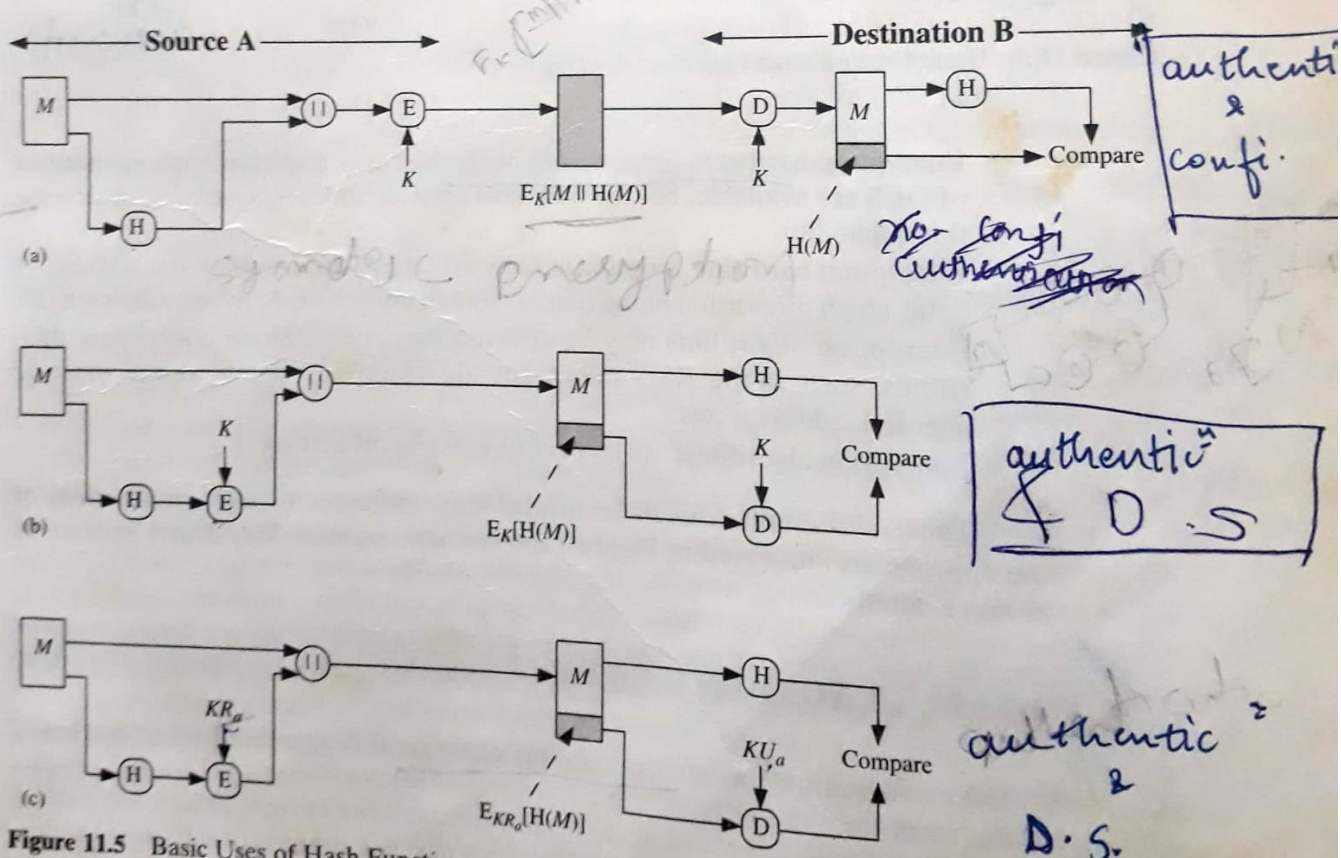

Figure 11.5 Basic Uses of Hash Function

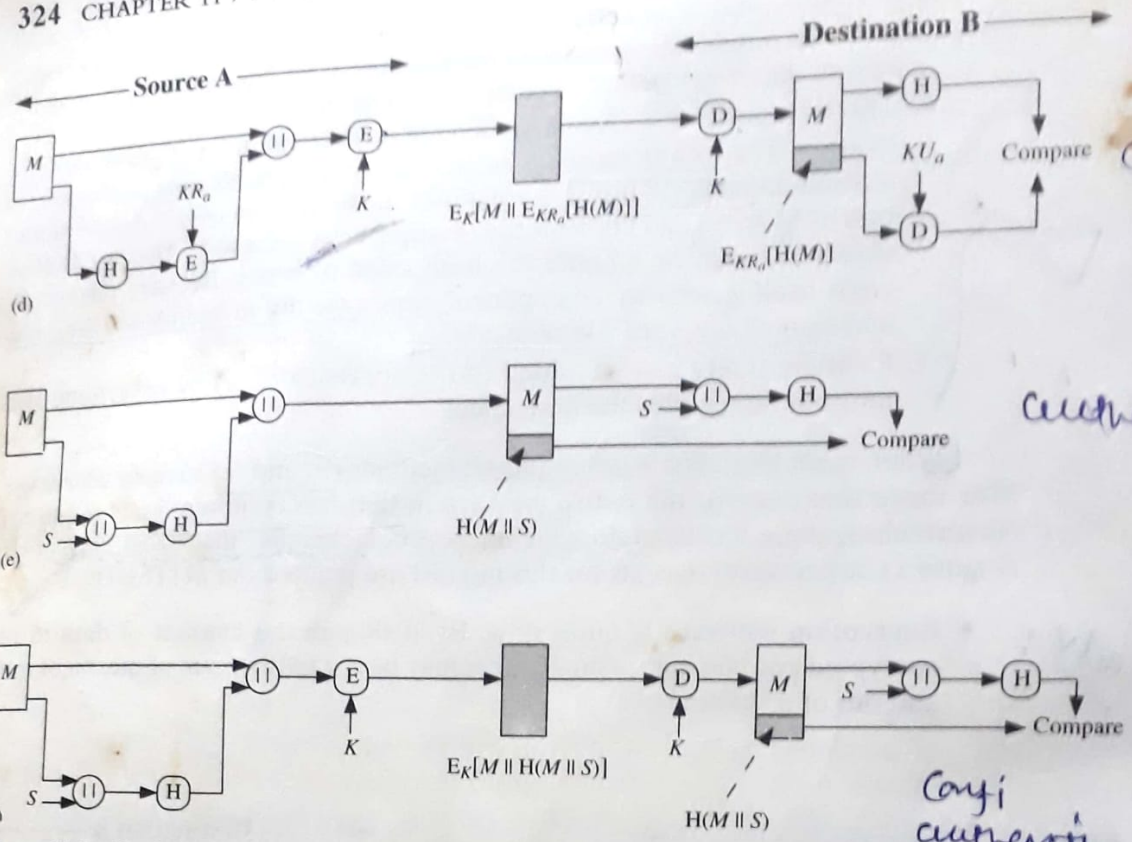

Figure 11.5 Basic Uses of Hash Function *(Continued)*

- Encryption hardware costs are not negligible. Low-cost chip implementations of DES are available, but the cost adds up if all nodes in a network must have this capability.
- Encryption hardware is optimized toward large data sizes. For small blocks of data, a high proportion of the time is spent in initialization/invocation overhead.
- Encryption algorithms may be covered by patents. Some encryption algorithms, such as the RSA public-key algorithm, are patented and must be licensed, adding a cost.
- Encryption algorithms are subject to U.S. export control.

Table 11.3 summarizes the confidentiality and authentication implications of the approaches illustrated in Figure 11.5. We next examine MACs and hash codes in more detail.

11.3 MESSAGE AUTHENTICATION CODES

A MAC, also known as a cryptographic checksum, is generated by a function C of the form

$$MAC = C_K(M)$$